# HUNTER VALLEY 99'ERS NEWS

TI 99/4A

## HOME COMPUTER NEWSLETTER

AUGUST 1987

TEXAS INSTRUMENTS

**Newcastle**

& The Hunter Region

Home Computer

USERS' GROUP

REGISTERED BY AUSTRALIA POST
PUBLICATION No NBG8023

The Secretary - HV'99ERS
6 Arcot Close, TARRO - NSW.
Australia - 2322

# YOUR COMMITTEE

PRESIDENT
Paul Mulvaney
26 Marmong St.,
MARMONG POINT 2284
Ph. 583623

VICE PRESIDENT
Alan Lawrence
35 Bayview St.,
WARNERS BAY 2282
Ph. 486509

SECRETARY
Albert Anderson
6 Arcot Close,
TARRO 2322
Ph. 662602
Viatel 496626020

TREASURER
Peter Smith
8 Glebe St.,
EAST MAITLAND 2322
Ph. 336164
Viatel 493361640

SOFTWARE LIBRARIAN
Alan Franks
822 Pacific Hwy.,
MARKS POINT 2280
Ph. 459170

PUBLICATIONS LIBRARIAN
Allen (Joe) Wright
77 Andrew Rd.,
VALENTINE 2280
Ph. 468120

EDITOR
Brian Woods
9 Thirlmere Pde.,
TARRO 2322
Ph. 662307
Viatel 496623070

PURCHASING CO-ORDINATOR
Bob MacClure
75 Deborah St.,
KOTARA SOUTH 2288
Ph. 437431

TECHNICAL CO-ORDINATOR
Gary Jones
53 Janet St.,
JESMOND 2299
Ph. 573744

SIGs CO-ORDINATOR
Brian Rutherford
9 Bombala St.,
DUDLEY 2290
Ph. 498184

# CONTRIBUTIONS

Members and non members are invited to contribute articles for publication in HV99 NEWS.

Any copy intended for publication may be typed, hand written, or submited on tape/disc media as files suitable for use with TI Writer (ie. DIS/FIX 80 or DIS/VAR 80). A suitable Public Domain word processor program will be supplied if required by the club librarian.

Please include along with your article sufficient information to enable the file to be read by the Editor eg. File Name etc. The preferred format is 35 columns and page length 66 lines, right justified.

All articles printed in HV99 NEWS (unless notified otherwise) are considered to be Public Domain. Other user groups wishing to reproduce material from HV99 NEWS may feel free to do so as long as the source and author are recognised.

Articles for publication can be submitted to the Editor, ALL other club related correspondence should be addressed to The Secretary.

# DISCLAIMER

The HV99 NEWS is the official newsletter of the HUNTER VALLEY NINETY NINE USER GROUP.

Whilst every effort is made to ensure the correctness and accuracy of the information contained therin, be it of general, technical, or programming nature, no responsibility can be accepted by HV99 NEWS as a result of applying such information.

The views expressed in the articles in this publication are the views of the author/s and are not necessarily the views of the Committee, Editor or members.

TEXAS INSTRUMENTS trademarks, names and logos are all copyright to TEXAS INSTRUMENTS.

HV99 is a non profit group of TI99/4A computer users, not affiliated in any way with TEXAS INSTRUMENTS.

# SECRETARYS REPORT

**FROM ALBERT ANDERSON**

Hello and welcome once again to the August '87 issue of the HV99 rag. Response to membership renewals this year has been really good with almost total renewal from our overseas and interstate people in already and about 70% local renewals. One very pleasing sidenote was quite a number of thank you notes and general hellos (that's G'DAY in Australian), that accompanied the the renewals. To those people us organisers would like let you know that we do appreciate your thoughts... thank you !!

We say welcome to some more new members this month in :-
Sid Murton from Hamilton
Neville Kennewell from Warners Bay
Les Tomlinson from Berowra Hts. in Sydney.
Duncan Pawley from Harbord also in Sydney.
Geoff Phillips from Kidman Park in the Adelaide area.

Thank you for joining the HV99'ers and please keep letting us know what your needs with the 4A might be. An article or two for the newsletter on your experiences with this machine would be nice to share also.

To those that as yet have not renewed their membership there is a reminder on the mailing label of the newsletter which says "LAST". This refers to the newsletter you are now reading and means that it will be *your last issue* until you renew. So if you haven't, a gentle reminder...DO IT!!!( please ).

To demonstrate the ongoing support the 4A machine has going for it here in Australia, some really great news comes from our friends up north in the Brisbane TIBUG group. Garry Christensen, the co-ordinator of TIBUG, has sent to us an invitation to participate in a TI-Faire that is being arranged to coincide with EXPO '88 in May of 1988. Now, think about it! what better place to escape the southern winter for while -- EHH! We know already that several HV99'ers are ready and eager to be part of it, so if any members wish to be part of it with us please let us know so we can start arranging things NOW!! Also to you TIsHUG'ers, TICHUG'ers, TIMES'ers, ATTIC'ers and TIUP'ers how about getting together with Garry and the TIBUGGERS (sounds like a rock & roll group, doesn't it?) and get this show on the road. Discussion between groups on arranging such an event has been thrown around for some time now, so lets grab this opportunity and make it happen. The letter from Garry appears in full elsewhere in the newsletter.

During this month we were also lucky enough to have a really TERRIFIC couple visit with us here in the Hunter. Charlie and Roslyn Chan from the Canberra TICHUG group brightened up a few nights for the lucky Novocastrians (that's people that have Newcastle as their home) that were able meet with them. Thanks to you both for taking the time to see the Hunter Valley and a few of your fellow HV99ers.

On the overseas front still many new programs and gadgets appear for the 4A. It seems RAM Disks are breeding and the support for them has sprung from everywhere. The overseas exchange newsletters have all the relative info on this and such things as Myarcs new HARD DISK/FLOPPY Controller. There is also news of an 80 column/ graphics card for the PE Box from DIGIT SYSTEMS in the US of A. and much, much more.. yes this is the TI99/4A I'm talkin' about here!! Joe Wright is the new publications man so get onto him for your share of this new info.

Still to do with overseas, HV99 are lucky enough to have some really enthusiastic members that have been asking for local members to write to them. As yet response to their requests has been dismal. Is this much to ask? You will be surprised what these people have to offer, let alone the personal contact. We have people from the USA, Canada, West Germany, Korea and now Italy. If you want a run down on any of them please let me know as I have had contact in some form with all of them. They are really great people and mad keen to share with us their 4A involvement.

While I'm on about contacts I still don't have VIATEL contacts from the Sydney, Melbourne, Adelaide or Brisbane groups. This is a very convenient message service and I would appreciate being able to leave and receive messages on this service.

Well its time to QE2 and get this to Ed. Oh! QE2 is a method that Funnelweb users use to get a document from monitor to paper.. nothing to do with the ocean liner bearing the same abbreviations.... bye now!!

Albert Anderson
4a4me

One outstanding event to keep in mind is the TI faire planned for May in Brisbane. See the letter from the Brisbane Users Group in this newsletter. We will keep you informed of the actual dates and times as we receive them. It would be nice if we could meet some of our overseas as well as interstate counterparts if they are coming to see EXPO 88. I saw the rather interesting building structures when I was in Brisbane some months ago. Following the success of the Melbourne Faire (what really happened in Melbourne? Our members still get smiles on their faces when it is mentioned) it is good to see another being organised.

Special thanks to Bob Carmany from North Carolina in the US of A. Bob has written the article on Speech in this issue and sent a program which form this months demonstration. It is good to see 'out-of-towners' getting involved in the demo's. I particularly liked Bob's segment on the foreign language capabilities of the TI, especially the 'Awestrine' bit. Thanks Bob.

Dont leave the choice of demonstrations to the committee, REMEMBER it is YOUR group. Let a committee man know what you want to see and we will arrange it. If you have a favourite program share it with the others. It does not have to be something you have written, it could be a program you have purchased, others may be interested in buying it as well but want to see it running first.

## *PRESIDENT'S*

It was very heartening to see new faces at the last meeting night. I would like to extend a welcome to all the new members. I spoke to a few of the new people and most said they did not know there was any support still around for our resilient Orphan. It makes you wonder how many more people there are out there who would benefit from knowing about the Users Group.

A warm welcome back also to our existing members. The future looks good with a number of activities already planned for the year ahead.

# SWAP REQUESTED

Ron Kleinschafer has a CorComp Mini System which includes a RS232 + power supply, only needs a disk drive & 32K card to become a full system. Ron wishes to SWAP it for a TI RS232 card + manual. If you are interested contact Ron on Ø68-293960 or Viatel 682939600.

# IN THE NEWS

A POTPOURRI OF INFORMATION

EDITED BY

JOE WRIGHT

The following news items have been taken out of newsletters which have been received in the last month.

They are all retyped and unedited.

From Worcester, Massachusetts. July newsletter called MUNCH. Writer "Dracula"

"Anne Dhein (outside Massachusetts she is THE expert on TI Writer [see last month's M.U.N.C.H. Newsletter]) has done it again. In an article in the Chicago UG Times she continues to write about TI Writer Graphics. She has written an EB programme that converts TI Artist instances into TIW files. Thus, you will be able to print graphics through FUNNELWEB or whatever TIW version you use. If it's as readily convertable as it sounds and comes out in DV80 format, this just might be the ultimate graphics/text programme for the 99. Henry enthusiastically exclaimed, "WOW! What's next!? I'm sure this was an unheard of or unthought item by TI's original designers! Did you ever think it was possible? Now who needs a graphics programme."

"Also can look forward to Peter Hoddie finally getting out FONTWRITER II. Peter's been so out straight with PRESCAN IT, XB BUG; MYWORD operations system for the GENEVE, his business with Corson, the travelling and writing he does, *that II* has been on the backburner. In the most recent LA Topics he says he should be finished within a few weeks."

From Memphis Tennessee. Mid South UG Newsletter TIDGITS, JULY. Writer Gary Cox.

"Corcomp of 2211-g Winston Road Anaheim CA 92806 has announced a TI/IBM Disk Copier which when used in conjunction with the Corcomp disk controller and 2 ds/dd disk drives will transfer text files from an IBM formatted diskette to a TI formatted diskette and visa versa using only the TI 99/4A. The main requirement for the IBM text data is that it must be in ASCII format. The programme is cartridge based, price of the product is unavailable."

From San Diego California. SDTI-SIG Newsletter June Edition. I could not find the writer's name to attribute the following. So instead I will thank the whole Group. Thank you.

" 'Fortran is now available for the TI,' was the announcement made by Rod Van Orden at our May meeting. Rod said a development package which includes a linker, compiler and debugger will be sold by Tenex Computer Express. See Tenex's Spring Catalogue. Rod commented, 'This is a real step forward for the TI-99/4A!'. The software's producer is listed as the LGMA Co. and the cost is $50. So what good is Fortran?. It's useful as a scientific development tool, notes Rod."

" Rod Van Orden again: Rod is compiling a c-99 manual. This will be a complete listing of documentation files and programme listings. It will amount to about 200++ pages of text and should be helpful to people who are struggling in learning our "small c" and want to converse with each other-- a sort of common reference."

" This may be a biggy. Tom Spillane of Digit Systems, a San Diego Co. producing a RGB Conversion kit (used for getting sharper video out of RGB monitors) has announced another product: An Advance Video Display Processor Card. What's that?

It will be a card that fits in the P.E. Box and provides more colors,more sprites, and 80 columns of text. Tom is using the same V9938 Yamaha chip that MYarc uses in

its 9640 and which Mechatronics uses in its 80 column card, to achieve this advance. It will also make it possible to control or display VCR tape segments in programmes. Home video fans will be able to install video overlays on their VCR tapes. Tom says an 80 column TI Writer will be made available. David Allen is working up demo software and is checking all existing TI software to see if it is compatible. To use this card, an analogue RGB monitor* with an 80 column rating is required. Expected cost will be $125. ( A Magnavox 8CM515 is an example of what is required.) So if the two other firms ( Myarc and Mechatronics ) are already providing this item, why should Digit? Tom says the card will fit in the P-Box whereas Mechatronics' is connected as a side-car attachment to the peripheral port. Also, that you have to buy the 9640 computer to get Myarc's. David remarks the graphics coordinates are simpler than the present TI chip the 9t9 now uses. Plain old x-y coordinates are used."

There is also a good review of SPAD XIII in this copy of SDTI-SIG.

From Brisbane, the land of the Banana Benders, July T.I.B.U.G. comes news of the release of a monitor interface for use with a the TI99/4A. They are selling the P.C. Board and instructions for $23:00. ( This includes postage).

The cost of components which you provide is between $40/$50. The unit is currently being used on a Phillips 34cm Model CP45 TV/MONITOR. The release states that the interface should operate with any set by other manufacturers which have an RGB input at TTL level. The address for the P.C. BOARD is;

TIBUG
C/O C.Bagley
4 Dakar Rd.
Algester 4115
Queensland.

Cheques payable to TIBUG.

I am sure that there is just as much news I have missed in the many other Newsletters we received. Good to see Stephen Shaw writing for Regularly for Ottawa. In the mean time keep your ear to the ground. If you pick up some news let me know so that it can be printed and SHARED in the TRUE spirit of a User Group.

One article which you should make a point of reading is in the April MICROPENDIUM. Mack McCormack has a description of his system and the use he has for each piece. VERY interesting!

Pete Smith's system with two RAMDISKS, and AMIGA monitor isn't too bad either, graphics are very impressive out of the little ole TI.

Oh! by the way while you have your ear to the ground keep a good look out for R.K'S missing black hole. You should be able to hear the truck idling. NO! it cannot run out of fuel in a black hole, or can it? Gotta go and tidy this room up before the cook walks in an catches me, else I might not live to write again.

# IMPORTANT ANNOUNCEMENT

A Magical Mystery Tour/Car Rally has been organised by Gary Jones for Sunday, 20 August, commencing at 9.30am.

The starting point will be the car park at Warners Bay High School, and the finishing point (if you should be so lucky!!) is ???

Please bring along your family, something for lunch - picnic only drinks & a sense of humour. (No Bar-B-que facilities available)

Take this opportunity to involve your family in the Group. After all, they have to put up with you pounding the keys at all hours of the day and night, don't they?

# BASIC SPEECH

BY

BOB CARMANY

I was asked by Paul Mulvaney if I would be interested in writing a series on using speech with the TI. Well, before I had given all of the ramifications of the decision full consideration, I told him that I would. I may have some significant afterthoughts once I start this!

Some years ago, TI offered the then newly-produced Speech Synthesizer as a "free speech" promotion with the purchase of a selected set of software packages. If I recall, you had to send in the package ends and they sent you back a Speech Synthesizer. The price on this "miraculous" piece of equipment dropped until now almost everyone has one attached to their console. It works fine for cartridges with built-in speech but there is little documentation for the independent use of the Speech Synthesizer. That leaves us at the present state of affairs!

This column is going to (hopefully) provide some information about using the Speech Synthesizer for text-to-speech and full utilization of the built-in vocabulary.

Besides the game cartridges that have built-in speech capabilities, there are three other cartridges that can access the speech capabilities of the Speech Synthesizer. The Speech Editor cartridge was one that was produced briefly and, for all practical purposes, is non-existant. So, we are going to concern ourselves with the two other cartridges that allow for direct speech access. The TE II cartridge is the first of these. It allows text-to-speech and other speech facilities as well. Let's start out by looking at it and what you can do with it.

To understand the TE II cartridge, we need to find out how the Speech Synthesizer generates the speech patterns that we hear.

The Speech Synthesizer processes speech by using the phonetic pronunciation of each syllable in the word. These phonetic "bits" are known as allophones. There are a total of 127 of these allophones that comprise the entire speech vocabulary. They follow a set of rules that govern how each letter or combination of letters is to be pronounced. Within the Speech Synthesizer, the entire resident vocabulary is stored as a series of allophones at a specific location. The TE II cartridge contains a listing of the 127 allophones that are the same as those used to create the resident vocabulary and are used for the unlimited text-to-speech capability. The interesting thing is that either the Speech Synthesizer or the TE II cartridge can be accessed to produce speech.

There is just one small problem with using allophones to produce speech phonetically. Not every word that you want to have spoken follows the "rules" for producing phonetic speech. As a result, you will have to mis-spell words from time to time to have the Speech Synthesizer pronounce them correctly. But, this does lead to another facet of text-to-speech that has not been fully explored. By phonetically spelling words, you can produce speech in foreign languages. For example, try the German equivalent of "Merry Christmas" --- phonetically spelled --- "FROLISHA VYNAKTEN".

In addition to this, you can vary the pitch and slope of the voice coming from the Speech Synthesizer to produce a variety from "Donald Duck" to a deep baritone. The documentation for the Speech Synthesizer contains the instructions as does Appendix C of the TE II manual. It follows the form:

// xx yyy where "//" is the preliminary command "xx" is the pitch (from Ø to 63) and "yyy" is the slope (from Ø to 255).

Another thing that you can do with the TE II cartridge is to add inflection symbols and stress points to your speech. The symbols "^" and "_" are the inflection symbols while ")" shifts the stress points within the word itself. Since the inflection symbols are non-alphabetic and cause a word break, they must precede the word in the PRINT statement.

Incidentally, TI tried to stuff the entire discussion of the TE II speech capabilities into some nine pages in Appendix C of the TE II manual. Considering the space alloted, they did a fair job.

The best way to find out what your Speech Synthesizer will do is to use the short allophone program in Appendix C and experiment with both the pitch and slope and use it to break the words down into their component allophones.

To utilitize the capabilities of text-to-speech in the TE II cartridge, you must first open a file (just like a printer). You will need to use this form:

```
OPEN #1:"SPEECH",OUTPUT
```

or

```
OPEN #1:"ALLOPHON",INTERNAL
```

Your decision on which one to use depends on whether you are using text strings (the first) or the allophone file (the second).

There is, however, a rather serious limitation to using the TE II cartridge for your speech programs. While it provides an unlimited speech vocabulary, you are forced to program in the very slow and cumbersome BASIC environment. If you could only use the programming power of XB along with the unlimited vocabulary of the TE II text-to-speech.

Now, we are going to look at the XB cartridge and what speech facility it has in it.

The XB cartridge was designed to access the "373 word" vocabulary that is built into the Speech Synthesizer. It does it with the CALL SAY subprogram. The form that it uses is really quite simple:

```
CALL SAY("HELLO")
```

or

```
CALL SAY(A$) where A$="HELLO"
```

This is fine if you are using single words but how about the phrases that are in the resident vocabulary. TI really screwed things up on this one --- they never adequately documented how to access phrases like "TEXAS INSTRUMENTS" with CALL SAY. The correct method has been published many times before but here it is "just one more time":

```
CALL SAY("#TEXAS INSTRUMENTS#")
```

You can make the phrases that you program more natural by using commas (","), periods ("."), or the plus sign ("+") between the words in the phrase. The comma inserts a slight pause, the period a longer pause and the plus sign links the words together so they sound more natural.

A second subprogram has been supplied with the XB cartridge to make speech access easier. It is the CALL SPGET routine. It provides access to the speech codes used for the individual words or phrases. It follows the form:

```
CALL SPGET("HELLO",W$)
```

If you then PRINT W$, you will get the speech codes used to produce the word HELLO. This can be valuable information for creating your own words by concatenating parts of several word strings.

The "373 word" vocabulary that is resident in the Speech Synthesizer is a serious limitation if you have a lot of words that aren't in it. Luckily, just after TI pulled the plug", they released a disk of Text-To-Speech A/L routines that allow unlimited speech capability from the XB cartridge without intruding on any of the programming space. There are three machine language routines on the disk that load into low memory expansion and allow text-to-speech facility. In a program, they would

be entered like this:

```
100 CALL INIT

110 CALL LOAD("DSK1.SETUP","DSK1.XL
AT","DSK1.SPEAK")
```

This loads the three routines into low memory expansion. Then, by using CALL LINK, we can access the unlimited text-to-speech.

```
120 CALL LINK("SETUP","DSK1.DATABAS
E")
```

This line will allow SETUP to load the database of allophones used to create the unlimited vocabulary. This statement only has to be executed once at the beginning of your program. The speech data will remain in memory until the expansion system is turned off. At this point, the text-to-speech system is ready to accept input.

To produce the speech itself, the following format is used:

```
CALL LINK("XLAT","HELLO",A$)
```

"HELLO" represents the text to be spoken (in this case a string constant) but string variables or string expressions can also be used. A$ is the allophone string into which the allophones which comprise the speech data for "HELLO" are placed. A second CALL LINK activates the speech facility.

```
CALL LINK("SPEAK",A$,xx,yyy)
```

A$ is the allophone return string from the previous CALL LINK statement. The value "xx" is the pitch (from 0 - 63) and "yyy" is the slope (from 0 - 255).

Remember, you can use the stress points and inflection symbols that we talked about earlier to produce a more realistic speech pattern as well.

The only limitation is that you are limited to 128 characters in the CALL LINK statement and the number of allophones returned cannot exceed 255. For longer phrases, simple divide the phrase into two or more parts.

You can also string allophones together with CHR$(x); where "x" is the allophone number. Remember to use "&" when you combine the allophones into a string.

The character codes between 249 and 255 have special meanings and can also be used in producing speech. Code 249 indicates a secondary stress point, code 250 functions as a sentence break and must be followed by two parameters (the first is the number of stress points before the secondary stress point and the second is the number afterwards), and code 251 indicates that a new default slope value follows. Code 252 indicates a new default pitch value, 253 indicates a rising pitch contour, and 254 is a falling pitch contour. Code 255, which is followed by a byte which gives the pitch parameter, functions as a temporary pitch modification for the next allophone.

I have sent along a demo program that illustrates some of the same basic concepts that we have discussed here in this column. Of course, you can also use speech in A/L and Forth but the A/L is better left to someone with the expertise of Tony McGovern and I understand that there isn't much of an interest in Wycove Forth. Maybe some other time . . .

*Bob is very keen to correspond with members of our Group, and is quite open to suggestions regarding material to be covered in his usual Random Bytes column. He is quite prepared to swap programs with other Users as well, so if you want a contact in the US, Bob is the man to get in touch with. His address is:*

*Mr. B. Carmany*
*1504 Larson Street,*
*GREENSBORO NC*
*27407 USA*

# BIRTHDAYS, ROULETTE, NEEDLES AND COIN TOSSING

This article by a member who continues in his endeavours to achieve anonimity.

In a casual encounter with the subject of probability, I found a few items of interest that proved suitable for experimentation in the computer environment. For those who may be interested in this kind of experimentation, here are some of the results.

One of the first items I came across was in a book which accompanied my TI-40 calculator. It was shown as an example of calculating the probability of two people having birthdays on the same day. Considering a year of 365 days (don't bring Leap Years into it) and a small group of people - say 23 - it may surprise some that the probability of two people having birthdays on the same day works out at roughly 50/50.

The solution is found by tackling the problem in reverse, ie by first working out the probability of the event NOT happening.

Consider one of the group (of 23). The probability that a second person will have a different birthday is (obviously) 364/365 - and that a third person's birthday will differ from the first two is 363/365 and so on to the 23rd person, when it's 343/365.

Straightforward multiplication (364/365)*(363/365)...*(343/365) gives the probability that everyone has a different birthday. As this works out to 0.492702765 or 49.3%, the probability that one person doesn't have a different birthday from the first person is therefore 50.7%. Other sized groups can be checked out with a short program to handle the multiplications.

```
10 CALL CLEAR
20 INPUT "HOW MANY PEOPLE? ":N
30 M=(366-N)/365
40 FOR L=(367-N) TO 364
50 M=M*L/365
60 NEXT L
70 M=INT((1-M+.005)*100)
80   PRINT   "PROBABILITY   WITH";N;
"PEOPLE IS";M;"%"::
90 GOTO 20
```

I won't go into the matter of 3 the same and 4 the same, as the calculations become quite complex, and I don't know the equations in any case.

So much for the theory - how does it work out in practice? Well, rather than wander around groups of people inquiring about their birthdays - a pastime no doubt fraught with danger! - a simulation using RND was knocked up.

```
10 CALL CLEAR
20 INPUT "HOW MANY PEOPLE? ":N
30 DIM A(100)
40 B=1
50 RANDOMIZE
60 FOR L=1 TO N
70 A(L)=INT(365*RND)+1
80 NEXT L
90 FOR K=2 TO N
100 FOR L=1 TO (K-1)
110 IF A(K)<>A(L) THEN 130
120 B=B+1
130 NEXT L
140 NEXT K
150 IF B=1 THEN 180
160   PRINT   :   "WITH  THIS  GROUP
OF";N;"PEOPLE";B;"HAVE  BIRTHDAYS";"
ON THE SAME DAY"::::
170 GOTO 20
180 PRINT "NO MATCH THIS TIME"::::
190 GOTO 20
```

Note that this is a how many times in a group trial, not just a 2 the same probability check. For a more effective trial the main program should be looped a number of times to obtain an 'average'. However this would be very time consuming - try just one group of 100 people to get an idea of how slowly the program runs.

To ward off the boredom, let's move along to Roulette, in which we have numbers 1 through 36 (18 odd and 18 even) and zero. In some less advanced countries there is also double zero.

Here is another field where probabilities (not necessarily actual occurrences) can be calculated with exactitude. And, once again, some 'reverse' calculations to find a nil result provide the probability figure for an event to occur.

As an example - let's consider a wager on the even numbers. For a wager on evens (or odds) the odds are evens.

It's obvious that for one spin of the wheel, the probability of an even number coming up is 18/37 and thus, conversly, of non-evens it's 19/37. So those figures are put into a formula to calculate the probability of evens coming up once - and only once - in (let's try) 5 spins of the wheel.

$$\frac{5!}{1!\,4!} * \left(\frac{18}{37}\right)^1 * \left(\frac{19}{37}\right)^4 = 0.169$$

and for twice only in 5 spins it's

$$\frac{5!}{2!\,3!} * \left(\frac{18}{37}\right)^2 * \left(\frac{19}{37}\right)^3 = 0.32$$

From these examples it's easy to see how the formula is applied. And to simplify matters, here is a program that does it all.

```
10 CALL CLEAR
20 INPUT "NUMBER OF SPINS? ":S
30 P=S
40 GOSUB 170
50 SF=Q
60 INPUT "NUMBER OF WINS? ":W
70 P=W
80 GOSUB 170
90 WF=Q
100 P=S-W
110 GOSUB 170
120 PF=Q
130 INPUT "LOOKING FOR  WHAT  CHANCE
(OUT OF 37) ":C
140   A=(SF/(WF*PF))*((C/37)^W)*  ((
37-C)/37)^P
150  PRINT  ::   "PROBABILITY   IS";
(INT((A+.0005)*1000))/10;"%" ::::
160 GOTO 20
170 Q=1
180 FOR L=1 TO P
190 Q=Q*L
200 NEXT L
210 RETURN
```

So for 5 spins (S=5) looking for no wins (W=0) on evens (C=18) the result is 3.6%. Therefore there's a 96.4% chance of a win (at least one - maybe more) in 5 spins. Moving away from evens to just 1 number (C=1), let's try 40 spins (S=40) for 1 win (W=1). The probability is 37%. And for two wins in 40 spins it's down to 20%.

Armed with the knowledge that there's a 96.4% chance of evens coming up at least once in 5 spins - and $31, a careful investor will make an initial investment of $1 on evens. If this is lost it's $2 next time and so on. After 5 spins, if evens still haven't appeared, the $31 is gone. Of course there's only a 3.6% chance of this happening. It's at this point that the daring investor (read Mad Gambler!) realises that after 6 spins the odds of failure are now only 2% and has come prepared with $63 to outlay...just to win $1!

Of course this example is a kind of worst-case situation and keen investors (inveterate gamblers), who are basically optimistic by nature, well know that it's only luck that counts in the long run and probability has nothing to do with it. There is also a formula variation to calculate the probability of achieving a certain amount of profit after a number of spins. This aspect has been bypassed as the results are usually quite depressing.

And so - on to the next sampling which concerns needles, and is known as Buffons needle problem. First consider a series of equally spaced lines - such as those on a sheet from a writing pad. Then 'picture' a needle being dropped onto it. What is the probability that the needle will fall across one of the lines? Who cares? Surprising, this theoretical problem has a practical application.

As with such probability problems there is a formula. However, in this case integral

calculus is involved. Enough said. Obviously the probability of a needle falling across a line will depend to a large extent on the length of the needle (compared to the distance between lines). Strangely enough if the length equals the distance between lines the probability calculates to an exact 2/PI.

To obtain an idea of the range of probabilities that can occur, the following trialling program provides some empirical results as a guide.

Inputs are for the number of trials and the unit length of the needle (relative to the distance between lines) eg twice the distance means input 2. The landing point for one end of the needle has a RND XL, YL location in the area between lines and a RND angle (A) at which it lies. The other end of the needle will therefore depend on this angle (using the sine) and it's unit length. As only vertical (Y axis) measurements are used, the XL, XM figures are not required. INTing the YL and YM locations will then indicate whether a line has been crossed or not.

```
10 CALL CLEAR
20 INPUT "HOW MANY TRIAL DROPS? " :T
30 INPUT "RELATIVE NEEDLE LENGTH?  "
:N
40 X=0
50 FOR L=1 TO T
60 RANDOMIZE
70 A=RND*8*ATN(1)
80 YL=RND
90 YM=YL+N*SIN(A)
100 IF INT(YL)=INT(YM) THEN 120
110 X=X+1
120 NEXT L
130 R=INT(100*X/T)
140 CALL CLEAR
150 PRINT "WITH NEEDLE LENGTH";N::
160 PRINT "AFTER A TRIAL OF";T;
"DROPS" ::
170 PRINT "LINE CROSSES WERE";R;
"%"::
180 GOTO 20
```

A few trials with N=1 should give a result somewhere near 2/PI (0.637) - 64%. For other relative needle lengths the approx. percentages should be somewhere around these figures:

```
N = .5  2  3  4
    32 84 90 93
```

If the excitement is proving too much at this stage, for a more mundane approach, it's not difficult to substitute a disc (such as a coin) for the needle.

Some readers (if any have mad it this far) may remember a sideshow game which involved tossing a coin onto a grid of lines forming squares. If the coin didn't fall across a line, you won a prize.

A variation of the preceeding program can be used to simulate the coin-tossing scam. For line 30, the input required will be "COIN DIAMETER? ". As calculations will now involve both X and Y axes, change line 80 on to:

```
80 YL=RND
90 YM=YL+N
100 XL=RND
110 XM=XL+N
120 IF (INT(YL)=INT(YM))*(INT(XL)=
INT(XM)) THEN 140
130 X=X+1
140 NEXT L
150 R=INT((1-X/T)*100)
160 PRINT :: "COIN WILL MISS A LINE"
170 PRINT :: "ONLY";R;"% OF THE
TIME" ::::
180 GOTO 20
```

A coin diameter of .25 will enable 16 coins to fit in a square. Make the diameter .2499 so that the edges of the coins won't actually touch any of the lines forming the squares. Before you try the program, what do you think the percentage odds will be that such a coin will miss the line?

There are a host of other subjects where probability formulas have been ascertained. I believe probability 'factors' have now been worked out for application to the probable results of particular sporting events. Even in cases where the formula are not known, the 99/4A is ideal for simulation trials. By the way, if you thought your 99/4A was pretty fast at calculations, try it out on 1000 needle drops! - you may feel the need for refreshments while you wait.

# EPSON UPDATE

PAUL MULVANEY

OPERATING INSTRUCTIONS:- Read instructions only when all else fails!

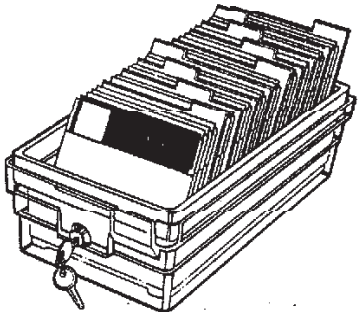
After nearly three years of using my Epson printer I finally resorted to reading the manual. One thing that I found was how to initialize the Special Character Generator. If anyone is like me and has not read the instructions and wants to see what the characters look like I have included a simple routine to print out the character ASCII number and the character. A sample run is included, I put the printer into wide mode so that they are easier to see.

```
100 REM  OPEN PRINTER CHANNEL
110 OPEN #1:"PIO"
120 REM  TURN ON EMPHASIZED MODE
130 PRINT #1:CHR$(27);"E"
140 REM  TURN ON WIDE PRINT MODE
150 PRINT #1:CHR$(27);"W";CHR$(1)
160 REM  TURN ON CHARACTER  GENERATOR
170 PRINT #1:CHR$(27);"m";CHR$(4)
180 REM  LOOP TO PRINT THE  CHARACTERS
190 FOR I=128 TO 159
200 PRINT #1:I;CHR$(I);"  ";
210 NEXT I
220 REM  TURN OFF WIDE PRINT MODE
230 PRINT #1:CHR$(27);"W";CHR$(0)
240 REM  CLOSE THE PRINTER  CHANNEL
250 CLOSE #1
260 END
```

Sample printout.

```
128 ┼   129 ┴   130 ┬   131 ┤   132 ├
133 ─   134 │   135 ┌   136 ┐   137 └
138 ┘   139 ▒   140 ■   141 ▬   142 ▪
143 ●   144 ○   145 ▲   146 ▼   147 ◆
148 ▪   149 ♪   150 ▫   151 ±   152 ▫
153 ↓   154 →   155 ↑   156 ↓   157 ×
158 ÷   159 ±
```

PROBLEM. If I want to use any of these characters in the text of a TI Writer file how do I input the information to the file?



# SOFTWARE LIBRARIANS NEWS

ALAN FRANKS

Well I am pleased to inform you that in the last month we have added another eleven disks full of programs to our library. Six disks full of music with interesting graphics printed on the disk labels came from Ken Gilliland in the States as a gift to our group. Ken said he wanted nothing in return for the disks but would be grateful for any letters he receives from anyone in the group, with comments on his programs which are freely available from the club library.

If you wish to write to Ken his address is :
543 RIVERDALE DR. #15
GLENDALE, CA 91204 USA

From Jack Sughrue one of our members in the States we received a disk with Jacks latest version of Funlplus. Also the Brisbane user group sent three disks of programs and John Paine, a club member from Mt..Druitt near Sydney, sent a disk of assorted programs.

For anyone needing blank disks they are only twelve dollars a packet at present. However there are only forty disks left out of the two hundred we bought last month and there is no guarantee the next lot will be as cheap, so if you need any now is the time to get them at the right price.

The demand for cassette tapes has dropped at present so I will make them up on request after present stock is sold. They can be programs from the disk of the month if they are suitable to run on cassette. While on the subject of the disk of the month I am always looking for volunteers to review it and you now get to keep the disk for your effort.

As I stated last month, anyone who wants any particular program or to browse through what's available in the library is quite welcome to give me a ring and arrange a convenient time.

# MODULE LIBRARY

WITH

RODNEY GAINSFORD

The module library has been in operation for a little over a year now and has currently 17 modules, 1 disk and 7 tapes. A list of the library is printed below. Don't forget all modules, tapes and disk are available for loan free of charge at the monthly meeting.

MODULES

The Attack
Blasto
Car Wars
Adventure
Mind Challengers
Star Trek
Dragon Mix
Meteor Multiplication
Addition & Subtraction
Reading On
Alligator Mix
Zero Zap
Tombstone City
Division 1
Rabbit Trail
Micosoft Multiplan

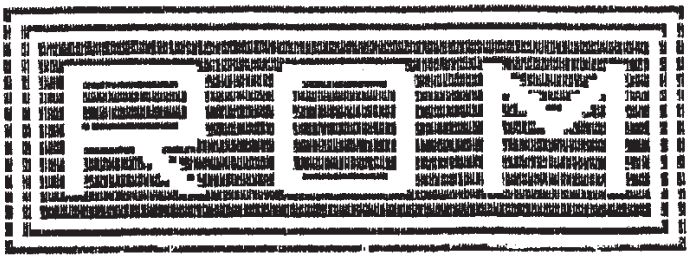
DISK

Microsoft Multiplan

TAPES

Beginners BASIC Tutor
Savage Island series
Adventureland
Pirate Adventure
Ghosttown
Voodoo Castle
Mission Impossible, The Count, Mystery Funhouse, Pyramid of Doom, Golden Voyage & Strange Oddessey

Could Ron Pratt, Helen Baverstock & Albert Hart get in touch with me as soon as possible please.



SERVING THE TI 99/4A HOME COMPUTER COMMUNITY

# READERS CORNER

**with Joe Wright**

Hello again! I hope the LOGO above attracted your attention. As promised last month I intend reviewing the contents of our Publications Library, starting with the Newsletters of User Groups with which the H.V.99'ers exchange. I am not doing them in any particular order, although doing this one first is no accident as you will read.

That LOGO above has been taken from the front of the ROM Newsletter of the USERS GROUP of ORANGE COUNTY. The Group is based in an area called Fountain Valley in County of Orange, California U.S.A. Orange County adjoins the southern boundary of Los Angeles County, between L.A. County and San Diego County. Fountain Valley is situated between Huntington Beach and Santa Ana, or in more specific terms, about 10 Kms south of Disneyland "as the crow flies". The address for correspondence is:

C/O Mr.J.N.Armstrong
1107 Primrose Dr.
ORANGE CA. 92668
U.S.A.

I have always held the opinion that the Newsletter produced by the H.V.99'ers is a window which allows people not within our Group to view us and make an assessment of our Group's strengths, weaknesses, versatility and cohesion. It follows that I view other Newsletters in the same light.

In the case of ROM I had only seen fleeting glimpses of it prior to becoming Publications Librarian. As a matter of fact I had accumulated some articles from ROM written by a Dale Loftis without fully realising where they had come from! For that I now make my apology and also thank ROM for these articles-come-tutorials.

To review ROM I decided to read all copies which were in the Library. The earliest we have is dated June 85 and the latest arrival at time of writing is June 87. I have not quite managed to get through to the last couple as yet but will do so in the next few nights.

At this point I can safely say the ROM is now on my MUST read list.

The Newsletter has well written and lively regular articles on Hardware ( K.Hamai ), Forth ( E.Raguse ) General T.I. BIT ( J.Sedlow ) in addition to these regular articles is a generous serving of other T.I. related information. Assembly Language, at which I am at the bottom of the learning curve, has been exremely well covered by the series Masters of Speed ( D.LOFTIS ). All of these articles are now inside the cover of my Assembly Language Tutorial folder.

The Forth articles I have drawn to the attention of Richard Terry, our Forth manipulator. I've read them but at this stage, since I have only a basic grasp of Forth, I find a lot of what appears in these articles (as I do with Richard's) quite bewildering. The articles are well written and have plenty of meat in them on which to chew.

The Hardware Series I found to be very interesting, & would like to particularly mention the April 87 article here. Anybody who has not as yet read any ROM then I would suggest that you start at this article, as soon as you can track me down. Maybe at the next General meeting hey! While on that article and so that you will have to read it I have a comment for Newt Armstrong the Editor of ROM. Good on Ya Mate!

All men have an unwritten right to have a room full of "GEAR", useful or otherwise. Here in Australia we call it the DOG HOUSE, that is where you can go when being yelled at to mow the lawn or some other useless job around the home. T.I. BITS Covers very thing related to the T.I. from the keyboard through to the wall socket. The articles are topical and once again well written.

I hope that I have covered enough of the Newsletter in this short space to have treated it fairly. I have not mentioned all the people who have contributed to the Newsletter but I do hope that I have done enough to arouse your interest. There are many other articles which you should take the time to read. To our friends of UGOC, if I have missed mentioning some people it has not been deliberate and not intended to offend.

A Newsletter cannot exist nor come together without a hard working often forgotten Editor, fully supported by the TOTAL membership of the Group. ROM is blessed with such people and the H.V.99'ers thank them all. So if you havn't read any ROM as yet then do so, I will include some copies in our next GREAT MAIL OUT.

Almost missed this bit! The Officers of the Group are listed on the front of each Newsletter. Seven of those names on the June 85 Newsletter still appear on the April 87 copy. Also sorry to see D. Loftis is unable to write regularly due to work commitments.

## NEW ARRIVALS

Well if I seemed to be complaining that last month had little to talk about then this month is a totally different story.
This is THEY; BAYOU BYTE (MAY), MELBOURNE ( APRIL/MAY ), BUG-BYTES BRISBANE ( JUNE/JULY ) SMART PROGRAMMER ( DECEMBER ), TIB-BITS ( JUNE/JULY ), OTTAWA ( MAY ), BITS, BYTES AND PIXELS ( JUNE ), MICROPENDIUM ( MAY/JUNE ), GUILFORD ( MAY/JUNE ), ATHENS ( MAY/JUNE ), ROM ( JULY ) & MUNCH ( MARCH/MAY/JUNE/JULY ).

This is current at the time of writing. As you can see there is a swag of information continually coming into the Group, to our local members, do yourself the service of using the Library, always remembering of course to enter what you have taken out into the borrowed book. ( It is an honour system which will only be wrecked by people who are not honourable.)

## THE GREAT MAIL OUT.

August is the month, I am just getting the final touches together and will hand over the mailing package to the volunteers at the August meeting. Hope you enjoy it as much as I do putting it together.

That's it from me, see you next month.
Joe Wright.

# AN EPROM PROGRAMMER

FROM

RON KLEINSCHAFER

CHAOS MANOR

Original circuit design by Heiner Martin (Germany).
Modification and Hardware Design by Ron Kleinschafer, HV99'ers.

THIS PROJECT IS NOT FOR COMMERCIAL REPRODUCTION

*Don't forget the usual disclaimer - if you follow the instructions contained in this article and stuff something up, TOUGH!!*

The Eprom Programmer presented here is a very versatile unit capable of programming any Eprom from 2K 2716's up to 16K 27128's. There are a few brands of Eprom that the unit cannot handle but these are rare, (see list below) & it would be possible to program these with some minor alterations to the power supply. Besides being able to program an Eprom for your TI it is also possible, with the support programs, to copy any Eprom for ANY device. As well, the code can be saved to disk so that other copies can be made as required, it can program a particular section of an Eprom then other code can be programmed in by "stepping" over those sections. This also enables the user to "burn down" (explained later) any particular bytes that may be wrong so that the whole Eprom may not have to be erased and reprogrammed again.

The unit is configured so that its CRU base address is at >1900, the space originally set aside by TI for their Eprommer. It is driven through the Module port and sends and reads data serially by the CRU out(A15) line and CRU in line which makes it very fast to Read, Verify, or Check that an Eprom is fully erased - in fact it can read an 8K Eprom in less than one half second, & to program an 8K*8 Eprom takes approx 7.5 minutes because of the required 50 Millisecond pulse to "burn" each byte.

The original design concept is very good and was designed by Heiner Martin a prolific hardware and software hacker from Germany. Extensive modifications have been carried out to make it as "goof proof" as possible, & I also decided to contain it into some type of box so that it could be transported fairly easy without causing too much damage. This makes it ideal for construction as part of a User Group type piece of equipment for loan out to members. Other design criteria have been taken into account including:-

1 :- The original unit derived power from the 5v line in the Module Port but with many consoles having the 32K memory on board and with the many other "gadgets" being fitted the console power supply is starting to be taxed to its limit.

2 :- The Printed Circuit Board was doubled sided with plated through holes and very "tight" using .015 tracks, making it impossible for the "kitchen sink" constructor to make one, so the aim was to produce a single sided PCB.

3 :- Replacement of a number of DIP switches that were confusing and could very easily cause "cooked" Eproms by not setting the correct switches. By using one 14 pin DIP "setup" socket and appropriate DIP plugs that, once selected for a particular Eprom, automatically sets up the Eprom with the correct read/write status and the correct Vcc and Vpp voltages.

4 :- Reduce the component count and select components that are easy to obtain. By using all new components the cost should be approx $30 (excluding the PCB)

Presented is a stand alone unit that incorporates its own power supply, is easy to construct and use, plugs into the Module Port and any TI99/4a owner could use it.

A quick BASIC rundown about the circuit diagram is as follows (see FIG 1):-
The 74LS685 sets the CRU base, the 74LS155 selects either a read or write operation, the 74LS299 collects the data serially and latches it into the Eprom for "burning", the 74LS251 does the reverse and reads the Eprom data and sends it out on the CRU-In line in serial format, the two 74LS259's generate the addresses as required. The power supply (FIG 2) uses a simple voltage doubler circuit with a LM723 precision voltage and current regulator so that the correct programming voltage of either 21v or 25v is automatically selected when the "set up" plug is inserted and the current is fixed to the maximum usually specified of 30Ma, the power supply also supplies the 5v required for all other functions. A further ideal modification would be to modify the PCB layout, fit an E/A Grom chip on board and do away with the Module socket - anyone got a spare ??

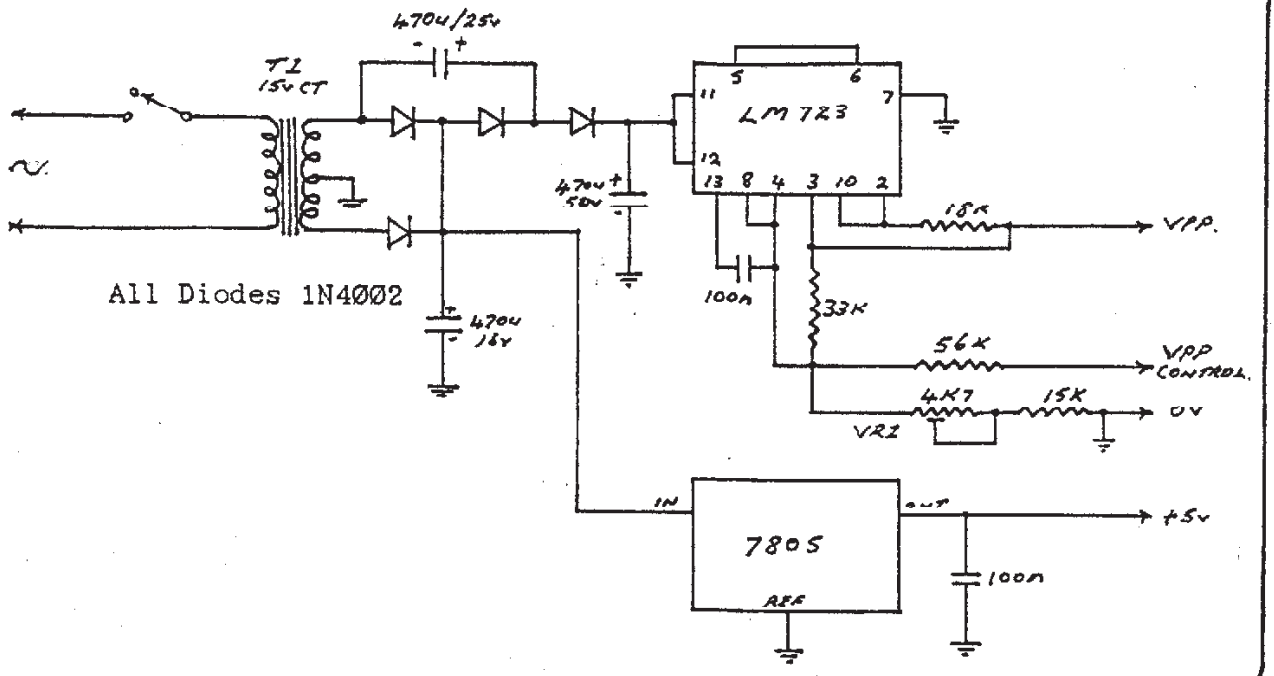
The PCB is made to fit as the "lid" of a large (60 * 113 * 196 mm.) Zippy box. The power supply is also encased as per FIG 4, the layout of the IC's not being very 'electronically elegant' insomuch that they do not all "point" in one direction, rather they are placed around the Eprom socket so that the track density is reduced. I don't think that it could be simplified any further.

On the PCB, probably the most tedious part of construction is that there are 35 wire links. These are necessary to achieve single sided PCB construction. They must be installed first because some run under IC's etc but from there on everything is straight forward. A standard miniature relay is used and is interfaced to the TTL circuit by an NPN transistor. The small switch is only used for programming 27128 Eproms, for the first 8K it is left at "normal", for the second 8K the switch is set to 128u to generate the A13 address and the Eprom start address of >0000 is selected from the software. Absolute care must be taken to make sure that all components are oriented as per the diagrams.

GENERAL CONSTRUCTION NOTES.
Most of the IC's are easy to get, but if you can't find the 74LS299 try GEOFF WOOD Electronics in Sydney (he has them all). The relay is a standard miniature 5v type (Dick Smith Cat. No. S7105), the Reset switch is an ultra miniature type (Dick Smith Cat. No. S1101), the Module socket is a UTILUX H271800 GD8. The unit is plugged into the

EPROMMER POWER

FIG 2.



Module Port using a somewhat modified Module and 36 core ribbon cable which is taken direct to the pins of the Module socket on the Eprommer PCB and bought out the front of the Zippy box, a small slot filed wide enough for the cable and deep enough to clear the Eprommers PCB at the front of the box is ideal. For more information on this see TERRY ROSS' article on Module port expansion in the HV99 Newsletter No. 6, but the switch and reset button is not used. The E/A Module is plugged onto the Eprommer with the TOP facing away from the Eprom socket. Sockets can be fitted for the IC's, the Eprom socket may be a standard 28 pin type but a Zero Insertion Force (ZIF socket) is recommended, on the bottom of the PCB is 4 jump wires as per the diagrams.

The transformer is bolted to the rear of the box with the power lead (with strain relief) exiting from the rear. The power supply PCB slides into the slots of the box vertically &only four wires are bought from the power supply to the Eprommer - these are øv, 5v, VPP and Control(VPP). It is a good idea to use a polarised plug and socket on these wires to make assembly easier.

The "setup" plugs are 14 pin DIL solder type. They are wired for each Eprom as per FIG 3, then suitably labled. When completed all that is required is to set the VPP voltage. With no Eprom inserted, plug in the "setup" plug for a 2764 Eprom, switch on the power and check the voltage on the pad on the PCB marked VPP (+) and GRND (-), adjust the trimpot VR1 until the voltage reads 21v, replace the "setup" plug for one made up for a 2716, 2532 or 2732, the VPP voltage should read 25v. While you are at it you may wish to ensure that each IC is getting the correct 5v at each pin as marked on the circuit diagram. If all is ok you are ready for a programming session.

Before continuing perhaps some basics on the misnomer of the term applied to Eproms being "burnt" should be discussed. Each "cell" in an Eprom is basically a Floating-Gate, Avalanche mode, MOS Transistor. A stored charge on the floating gate determines whether it effectively stores a "1" or "Ø". When an Eprom is exposed to Ultra Violet light it removes this stored charge thus making the cell become a "1", IE:- it is not "burnt" but simply its conductive status is changed.

On the program support disk are six files. These are DEBUG, EPROM/L, EPROMC, MEMSAVE, SAVE and CONVERT (XB). The authors of some of these programs are unknown and some have been considerably changed. DEBUG is used to move Program files into desired memory locations and is also used to save copied Eproms code to disk. EPROM/L is the TI BASIC loader for the EPROM program, SAVE is the TI BASIC loader for the programs to save any Eprom code to disk, CONVERT is an XB program for changing DIS/FIX 8Ø UNCOMPRESSED files that are AORG'd at say >6ØØØ or >4ØØØ that would not normally load. It changes the LOAD ADDRESSES for each file and at the same time changes the CHECKSUM TAG so that the files will load with a CALL LOAD(DSKx.xxxx) but the data actually loaded is unchanged from the original file. Incidentally, the data can be later saved back to disk as a Memory Image Program file to save disk space if you wish.

A typical Programming session would go like this.
Plug in the Eprommer, Plug the E/A Module onto it, select the "setup" plug for the particular Eprom you will be using (for this example all references will be for a 2764 Eprom and an 8k file),put the Program Disk in Drive 1.

TO COPY AN EPROM.
Load the Eprom to be copied
OLD DSK1.EPROM/L <enter>
RUN <enter>
CALL LINK("EPROM") <enter>
Screen display :-
Eprom Burner (E/A)

16k Eprom (Y/N) press N
Ram Start Address input >CØØØ
Ram Last Address+1 input >EØØØ
Eprom Start Address input >ØØØØ

Read Ff Prog V'fy Back End

Turn on the Eprommer power and press the Reset button. NOTE :- although the Eprommer auto resets on power up it is a good idea to get used to it.

Press R to Read.
Install an erased Eprom.
Press the Eprommers RESET button.
Press P to program.
When finished press the RESET button then press V to Verify, if all is Ok then "Okay" will be displayed.

The GREEN LEDS on the front of the Eprommer light during a Read, Verify or Ff check, the RED LEDS light during Programming.

TO SAVE THE CODE TO DISK.
Press E (returns to Basic)

```
OLD DSK1.SAVE              <enter>
RUN                        <enter>
CALL LINK("DEBUG")         <enter>
U                          <enter>
N C000,V1006,2000          <enter>
M V1000
xx enter 00                <Space bar>
xx   "   00                  "     "
xx   "   20                  "     "
xx   "   06                  "     "
xx   "   C0                  "     "
xx   "   00                <enter>
M 267C                     <enter>
shows 0000 change to 2006    <enter>
M 2684                     <enter>
shows 2E20 change to 2E42    <enter>
R                          <enter>
W shows 0000 do 8300                <Space b.
P shows 0000 do 2690                <Space b.
S (should show 0000)                <enter>
```

Press Q and the code will be saved to DSK1 under the filename "B" (you can change this name with a Disk Manager to whatever you wish). All that has been done is to move the code into VDP Ram, add a Header and the code is saved in TI's memory image format, and can be used later to burn another Eprom, no matter what the Eprom is used in. In this example the 2006 is the Program length plus 6 bytes for the header, and it will load at >C000. Suggested reading is TI's Debugger Commands.

TO PROGRAM AN EPROM FROM A DISK FILE.
If the file is DIS/FIX 80 and not AORG'd at >4000, >6000, LOW/MEM or any other unusual Address, then just CALL INIT :: CALL LOAD("DSKx.xxxx") & it will load at its designated memory area (NOTE:- The file must load into HIGH-MEM otherwise it will overwrite the Eprom Program). You can then RUN the Eprommer Program and burn your Eprom from that address. If the file is Memory Image then RUN the Eprommer Program and OLD the filename at the screen prompts, this will load into VDP Ram at approx >700. A Warning of "I/O ERROR 50" will be displayed but it is already loaded so you just ignore it and continue. By using Debugger you can locate and move the file to High Memory then CALL LINK the Eprommer Program and continue from there.

If the file is AORG'd at >4000 or >6000 then first change it by running the XB program CONVERT, it will be saved back to DISK as a new file with a "*" appended to its filename. You can designate the memory area you want the file to load at (NOTE :- ALWAYS USE HIGH-MEMORY because the Eprom Programs use the LOW-MEM area), it will then load with a CALL INIT :: CALL LOAD. It is a good idea to always save your DIS/FIX 80 files back to disk as Memory Image files because it is easy to find the length of the file when used again - the first two bytes are either 00 00 or FF FF, (the latter only if you are programing a file with another part of the program to follow). The second two bytes are the program length - this can be noted and saves time programing.

BURNING DOWN.
Sometimes if a mistake is made in the code it may be possible to alter the required bytes without erasing the entire Eprom and starting again. This process is known as "burning down" the main criteria being that the cells you wish to change must contain "1's". The following table shows which can be "burnt" down and those which cannot.

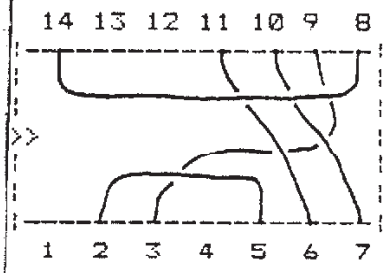
EPROMS NOT SUPPORTED.
(As far as known)

| Number | Brand. |
|---|---|
| TMS2716 | (TI) |
| 2764A | (Intel,AMD) |
| 27128ADC | (AMD) |

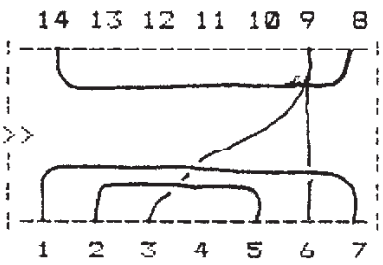
These Eproms require 12.5v VPP or Dual supply rails. If you have an Eprom you are not sure of check the manufacturers data sheets.

## WIRING THE SETUP PLUGS
(TOP VIEW)

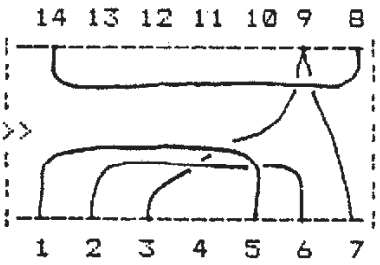
FIG 3.



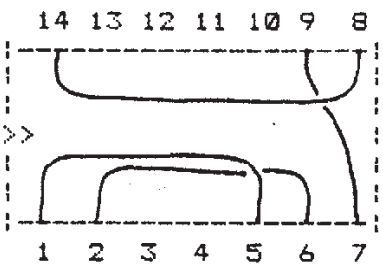

2716/2516(TI)



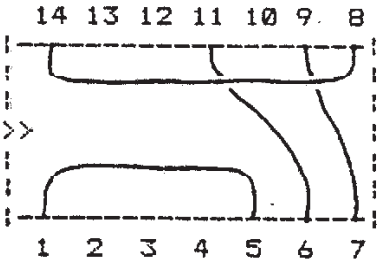

2532




2732

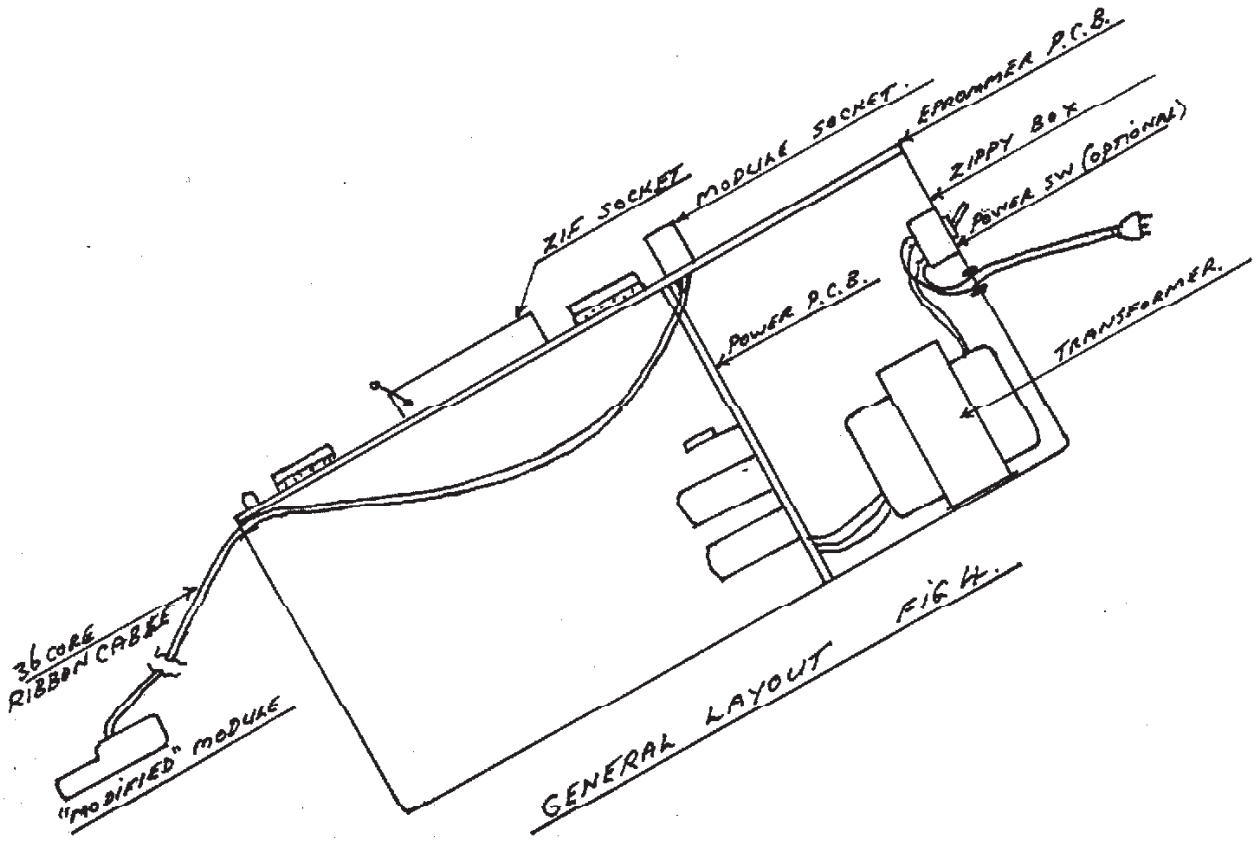



2732A




2764/27128

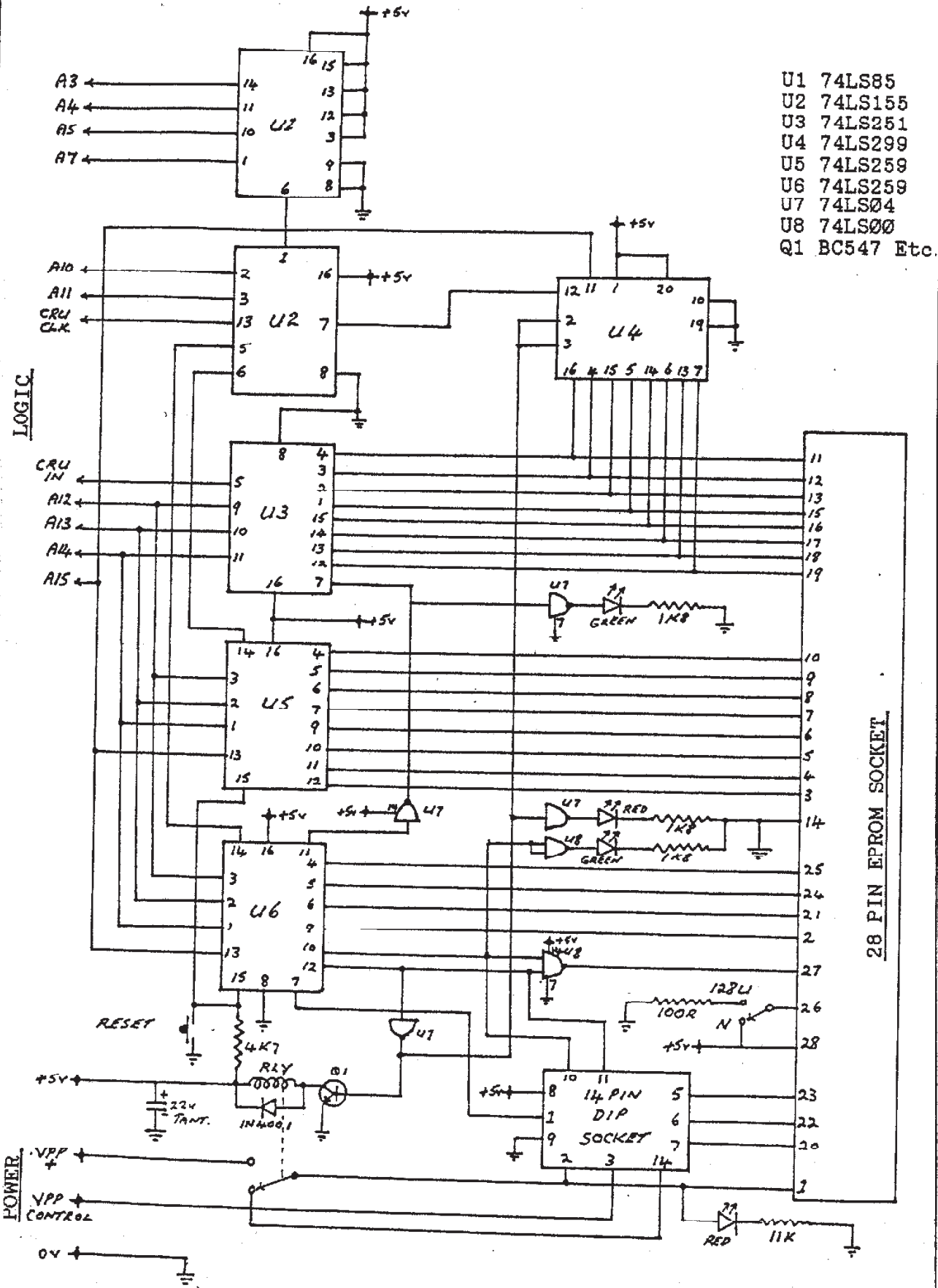
A suggested method of labeling the plugs when "proved" is to fill the top over the wires with Epoxy Resin then stick each label on, when dry cover the label with clear Araldite



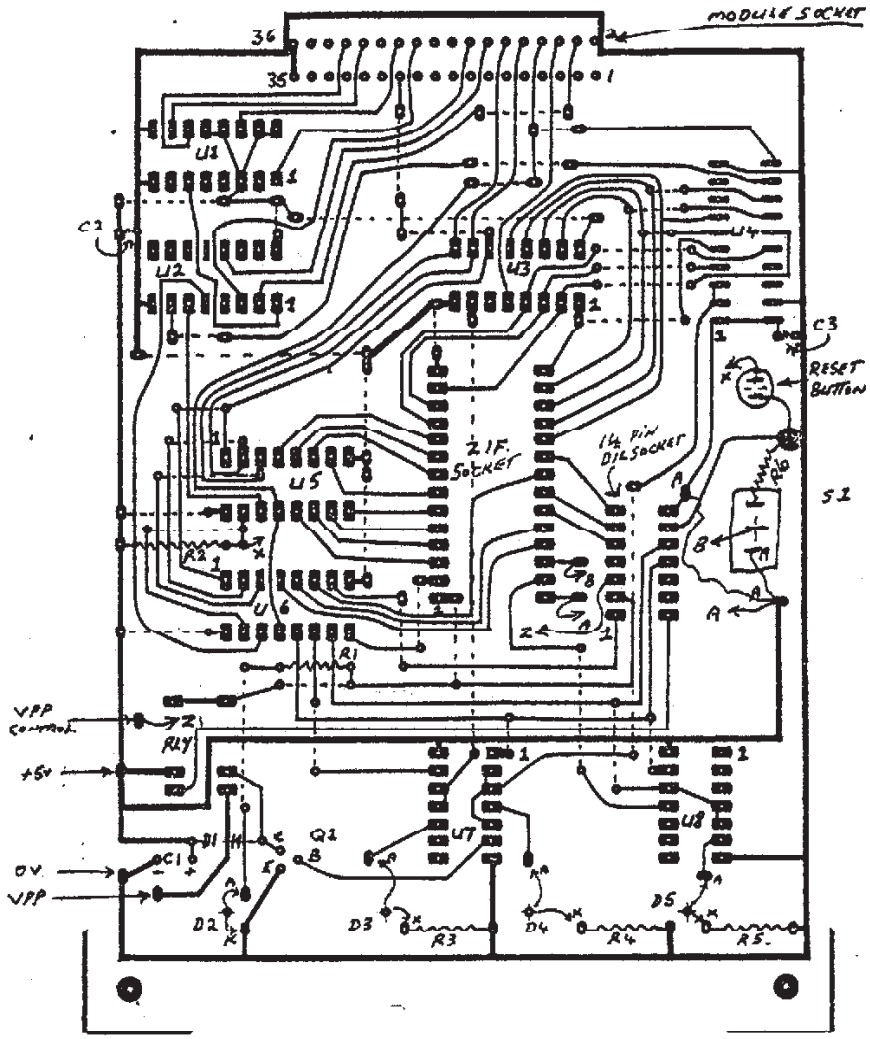

GENERAL LAYOUT FIG 4.

# EPROMMER LOGIC CIRCUIT

FIG 1



PCB (UNDERSIDE)




DOTTED LINES ARE LINKS ON TOP OF THE PCB.
S1 FORWARD TOWARDS THE RESET BUTTON IS "NORMAL".
JUMP WIRES ARE A/A B/B X/X Z/Z

| | |
|---|---|
| D1 | 1N4001 |
| C1 | 22uF TANTALUM |
| C2-3 | .1uF MONOLITH |
| D2-4 | 3mm GREEN LEDS |
| D3-5 | 3mm RED LEDS |
| R1 | 11K |
| R2 | 4K7 |
| R3-5 | 1K8 |
| R6 | 100R |

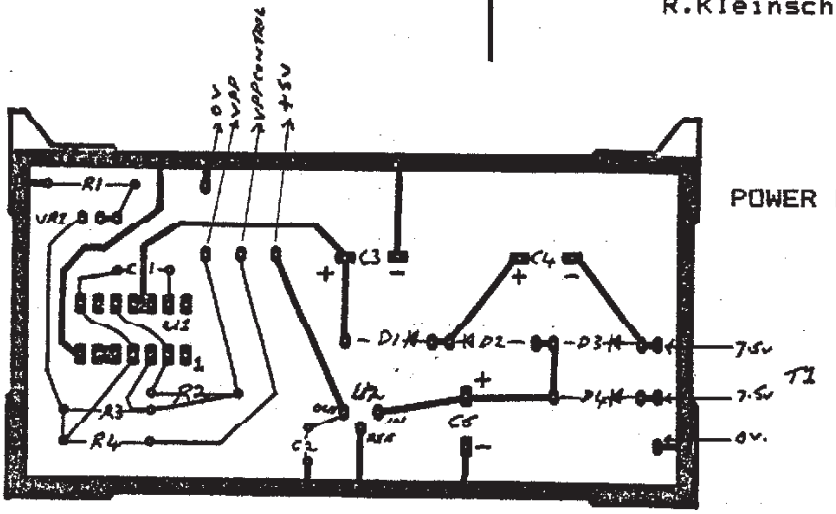
Value. Can be "burnt" down to

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| Ø | Ø | | | | | | |
| 1 | Ø | | | | | | |
| 2 | Ø | | | | | | |
| 3 | 2 | 1 | Ø | | | | |
| 4 | Ø | | | | | | |
| 5 | 4 | 1 | Ø | | | | |
| 6 | 4 | 2 | Ø | | | | |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | Ø |
| 8 | Ø | | | | | | |
| 9 | 8 | 1 | Ø | | | | |
| A | 8 | 2 | Ø | | | | |
| B | A | 9 | 8 | 3 | 2 | 1 | Ø |
| C | 8 | 4 | Ø | | | | |
| D | C | 9 | 8 | 5 | 4 | 1 | Ø |
| E | C | A | 8 | 6 | 4 | 2 | Ø |
| F | Any value | | | | | | |

To sum up, the unit may not be as fast as the Mechatronics Eprommer with their fast programming mode and doesn't have fancy buffers etc but it is just as flexible and the bottom line is that it sure is one hell of a lot cheaper.

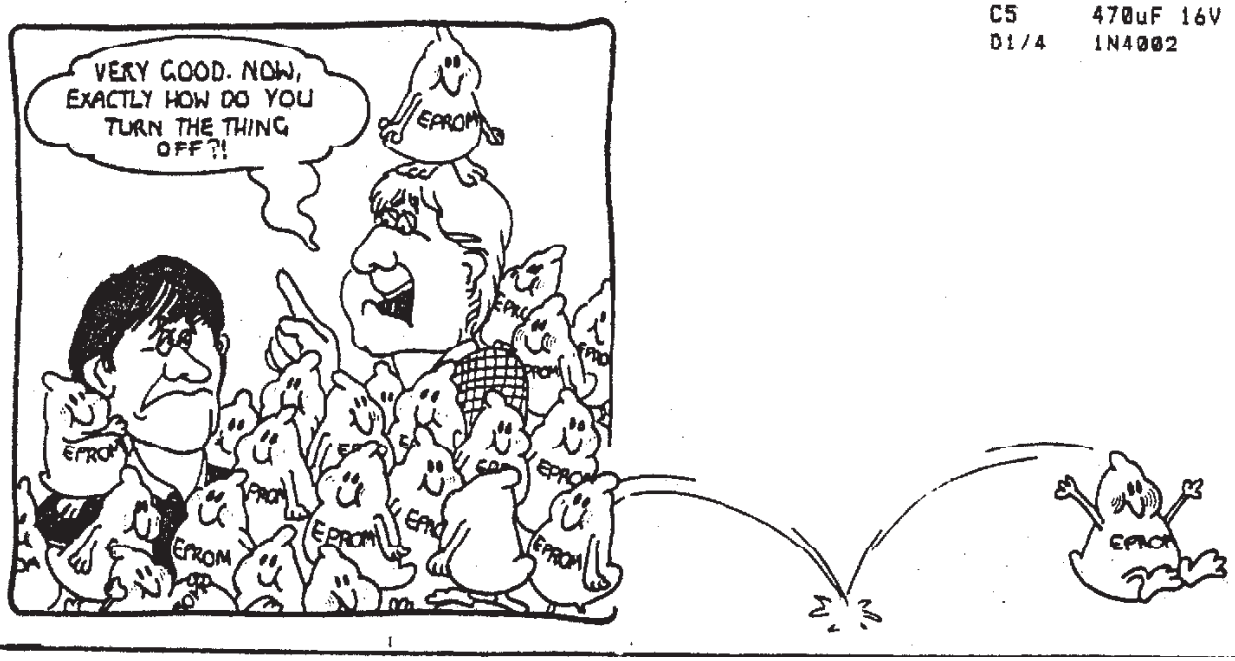
For anyone that may be interested the original taped up artwork is available on loan for Photographic Process from the Secretary, HV99ers, Mr Albert Anderson. Alternativly, the PCB could be made up by the method of "drill first, etch last" as published in the HV99ers Newsletter October 1986. Any further queries may be directed to me on Phone Ø68-293960.

R.Kleinschafer.




POWER PCB (UNDERSIDE)

| | |
|---|---|
| U1 | LM723 |
| U2 | 7805 |
| R1 | 15K |
| R2 | 8R |
| R3 | 33K |
| R4 | 56K |
| VR1 | 4K7 TRIMPOT |
| C1/2 | .1 GREENCAP |
| C3 | 470uF 50V |
| C4 | 470uF 25V |
| C5 | 470uF 16V |
| D1/4 | 1N4002 |



# INVITATION

COME UP AND SEE ME IN MAY.

This letter was received from the BRISBANE USERS GROUP.

36 Henzell St,
Kippa-Ring. 4020
4th July, 1987.

I would like to take this opportunity to invite all the members of the Hunter Valley Users Group to attend the TI-Faire that is to be held in Brisbane in May next year.

We have invited many of the major manufacturers and retailers who cater for the TI-99/4A comouter and to this date the response has been favourable. A list of the business houses attending the Faire will be forwarded at a later date. It is hoped that on display will be most of the latest hardware and software available for the TI computer and for the Myarc 9640 computer.

I would also like to invite the Hunter Valley Users Group to set up a table and display any software that the members have written and any hardware upgrades that have been pioneered through your Group. Representation from every Users Group in Australia would help demonstrate to our American counterparts that support for the TI computer is still strong in Australia.

There are many good programmes that have been or are being developed in Australia as well as some innovative hardware. The Faire will provide an opportunity to share the knowledge with the other users in the country. Central meetings of the interstate users are too few and far between. At present the only official contact between Users Groups is through the exchange of newsletters. The continued success of the TI computer requires co-operation between the Users Groups and the free flow of information.

We are also inviting computer retailers throughout Australia to attend the Faire. A strong showing at the Faire may induce some interest from computer retail businesses in the market for software and hardware associated with the TI-99/4A. I am tired of having to send to the United States to purchase programmes. There is a need for a distributer in Australia.

Aside from seeing the 'latest', there is another reason to come to Brisbane at that time. The Faire is timed to coincide with the opening of the International Exposition EXPO 88 several weeks before. While there may be other TI-Faires in the future, another EXPO will not be hosted in this country in our life-times. This is a once in a life-time chance and you will have the opportunity to attend the TI-Faire as well.

Once again let me stress that the continued success of the TI computer and compatibles will only be through the support of the users. Your support is vital. We can only guess, but in excess of 20,000 TI-99/4A computers must have been sold in Australia. Less than 750 users are members of Users Groups. There is an awful lot of people who are interested in the computer but are not members, and they must be interested because TI-99/4A computers are not being sold. These people probably know nothing of the enormous leaps that have been made in the field. Promote this Faire in your own area and the membership of the Users Groups must increase.

The work that all users put in today will benefit them in the future. I ask that the Hunter Valley Users Group support this Faire. Perhaps you would consider publishing this letter in your newsletter. The invitation is open to all the members as they are the ones who will benefit.

The invitation remains open to all who may be visiting EXPO 88 later in the year. Drop in and say 'Hello'. Interstate visitors are always welcome.

I wish the Hunter Valley Users Group success in the future and hope to see many members at the TI-Faire next year.

Yours Faithfully,
Garry J Christensen
Co-ordinator
TI - Brisbane Users Group

# TI BASIC CLASS NOTES

by

paul mulvaney

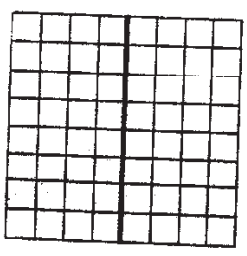
The character definition capabilities of the TI computer is one of the things that has endeared the Orphan to many programmers. With the CALL CHAR subprogram you can define your own graphics characters. It is possible to redefine from ASCII 32 to 159. The subprogram format is;
CALL CHAR(*char-code,"pattern-identifier"*)
The char-code is the ASCII code number of the character you wish to alter. The pattern-identifier is a 16 character string which specifies the pattern of the new character. To understand the make-up of the pattern-identifier it is useful to know a little about the Hexadecimal numbering system.

The Hexadecimal numbering system, often referred to as Hex, consists of a number system with the base 16. It would seem logical that the numbers used in Hex would be 0 through 15, but in reality only the numbers 0 to 9 are used and then the letters A to F. Each Hexadecimal digit can be represented by four Binary Digits. In the Binary numbering system the base 2 is used and each digit has a value to the power of 2.
For example $2^3 = 8$, $2^2 = 4$, $2^1 = 2$ and $2^0 = 1$

By turning 'ON' combinations of these digits we can form the character patterns to define the shape we place on screen, and hence the Hex code for the CALL CHAR instruction. In the Binary code a 1 is ON and the value is added, a 0 is OFF and the value is not added.

| Binary | | | | Character Pattern | Hex | Decimal |
|---|---|---|---|---|---|---|
| $2^3$ | $2^2$ | $2^1$ | $2^0$ | | | |
| 8 | 4 | 2 | 1 | | | |
| 0 | 0 | 0 | 0 | | 0 | 0 |
| 0 | 0 | 0 | 1 | | 1 | 1 |
| 0 | 0 | 1 | 0 | | 2 | 2 |
| 0 | 0 | 1 | 1 | | 3 | 3 |
| 0 | 1 | 0 | 0 | | 4 | 4 |
| 0 | 1 | 0 | 1 | | 5 | 5 |
| 0 | 1 | 1 | 0 | | 6 | 6 |
| 0 | 1 | 1 | 1 | | 7 | 7 |
| 1 | 0 | 0 | 0 | | 8 | 8 |
| 1 | 0 | 0 | 1 | | 9 | 9 |
| 1 | 0 | 1 | 0 | | A | 10 |
| 1 | 0 | 1 | 1 | | B | 11 |
| 1 | 1 | 0 | 0 | | C | 12 |
| 1 | 1 | 0 | 1 | | D | 13 |
| 1 | 1 | 1 | 0 | | E | 14 |
| 1 | 1 | 1 | 1 | | F | 15 |

The screen character is made up of an 8 x 8 grid. If we divide the grid down the centre we have two columns of four digits. Each column has a Hex value corresponding to the Binary representation. Each row therefore has a left (A) and a right (B) Hex value. The 16 values form the pattern identifier or Hex Code for the character. If less than 16 values are entered the remainder are assigned the default value of 0.




| | A | B |
|---|---|---|
| Row 1 | | |
| Row 2 | | |
| Row 3 | | |
| Row 4 | | |
| Row 5 | | |
| Row 6 | | |
| Row 7 | | |
| Row 8 | | |

The CALL CHAR subprogram only defines the character, to place it on the screen requires CALL HCHAR< CALL VCHAR or PRINT. These commands have been discussed previously. For examples of the use of CALL CHAR open your Users Reference Guide to page II-78. Remember if you redefine ASCII 32 the screen will fill with the new character as 32 is the space character.

To make your new character more effective you can alter the colours of the 'ON' and 'OFF' bits of the character. The 'ON' bits are referred to as the foreground colours and the 'OFF' bits are the background colour. The default colours are Black and Transparent. The transparent background allows the screen colour to show through. The characters are divided into sets of 8 for the CALL COLOR subprogram, all eight characters in the set are changed to the specified colours. The format for the subprogram is;
CALL COLOR(*char-set,foreground-colour,background-colour*)

The following table shows the colour numbers and the colour sets.

| Code | Colour | Set | Characters |
|---|---|---|---|
| 1 | Transparent | 1 | 32—39 |
| 2 | Black | 2 | 40—47 |
| 3 | Medium Green | 3 | 48—55 |
| 4 | Light Green | 4 | 56—63 |
| 5 | Dark Blue | 5 | 64—71 |
| 6 | Light Blue | 6 | 72—79 |
| 7 | Dark Red | 7 | 80—87 |
| 8 | Cyan | 8 | 88—95 |
| 9 | Medium Red | 9 | 96—103 |
| 10 | Light Red | 10 | 104—111 |
| 11 | Dark Yellow | 11 | 112—119 |
| 12 | Light Yellow | 12 | 120—127 |
| 13 | Dark Green | 13 | 128—135 |
| 14 | Magenta | 14 | 136—143 |
| 15 | Gray | 15 | 144—151 |
| 16 | White | 16 | 152—159 |

# TI WRITER TIP

*This article, by Jane Laflamme, originally appeared in the October 1986 issue of the newsletter of the Ottawa TI 99/4A Users Group.*

I have found that the .TL command seems one that is misunderstood or avoided. To me the Transliteration command is one of the most powerful formatting tools in the TI-WRITER package. Perhaps the best way to describe it, for you BASIC programmers, is as a "string variable". If you wish to set your printer in BASIC, you might do it this way:

```
100 OPEN #1:"PIO"
110 A$=CHR$(27)&CHR$(83)&CHR$(0)&CHR$(15)&CHR$(27)&CHR$(49)
120 PRINT #1:A$
```

What I have done with the BASIC statement 110 is send characters 27,83 and 0 to set my printer, an SG-10, to superscript mode: character 15 will condense the print and 27 and 49 will set it to 10 lines per inch. Once the variable A$, line 120, has been sent the printer will be set in that mode until it is turned off. The .TL command can be used in the same way through your word processor. Of course, TI-WRITER will open and close the file link to the printer for you automatically so you don't have to worry about line 100. The equivalent .TL statement would read this way:

```
.TL 60:27,83,0,15,27,49 (cr)
< (cr)
```

The character I have used for the "variable" is 60 or <. When the Formatter encounters the <, as instructed it will send the "string" to the printer. Again, once set, the printer will stay that way until it is turned off. It therefore leaves that character open for use within that particular document for another use. Or you can .TL it back to itself, .TL 60:60, and print it as itself in the same document. You therefore can use any characters you wish after they have done their job. I have used character 60, but you could use any character you wish. A .TL command should be on a line by itself and not exceed the 80 columns.

# Adventurers' Corner

WITH "THE ADVENTURER"

## rodney gainsford

Brought to you from The Adventurers Guild, the original and still the best...

### ADVENTURERS CORNER

After missing a few articles due to exams and illness, we have bounced back with a bumper article so now, let's jump straight in...

### ENCHANTER

You need to find 13 Scrolls to achieve your goal. You can find out what it's like to be sacrificed and talk to frogs and turtles.

- Like the sign says, Other Way

| SPELL | FUNCTION | LOCATION |
|---|---|---|
| Gondar | Quench open flame | Library |
| Izyuk | Fly like a bird | Falling |
| Guncho | Banish to other plane | Terror Room |
| Filfre | Create gratuitous fireworks | Map Room |
| Melbor | Protect caster from evil beings | Box |
| Kulcad | Dispel magic spell | Engine R |
| Frotz | Cause something to make light | Book |
| Nitfo | Talk with beasts in their language | Book |
| Blorb | Protect small obj | Book |
| Gnusto | Write spell to bk | Book |
| Rezroz | Open locked or enchanted object | House |
| Cleesh | Change creature to small amphibian | Swamp |
| Krebf | Repair wilful damage | Forest |
| Exex | Make things move with speed | Cell |
| Ozmoo | Survive unnatural death | Gallery |
| Zifmia | Summon a being | Egg |
| Vaxum | Make hostile Create friend | Bedpost |

- Show the adventurer the egg and he will follow you
- Guncho krill
- Gordar dragon
- Vaxum being
- Ozmoo self when tossed in cell

### MYSTERY FUN HOUSE

- To get in, get branch in parking lot. Chew gum. Go buy a ticket.

### PIRATE ADVENTURE

-Take book. Read book. Open book (twice). Go passage. Get bag and torch.

- On ledge wear sneakers and with book, say YOHO.

### WITNESS

- Look for changes before and after Linders departure.
- Responses to a given question may vary if more than one party is present at the time.

### ZORK III

- Be trusting.
- Fight shadowy figure. When defenceless, take hood and cloak...don't kill him.
- Save game and take lamp into lake 'for fun'.

### GOLDEN VOYAGE

- Don't steal.
- Buy everything with King's gold

**SUSPENDED**

- To minimize casualties, send a robot to weather control, turn second dial to zero

**STARCROSS**

Scoring Summary
(25 points for each)
Reaching Artifact's airlock
Black Rod (entering artifact - press 4th)
Yellow Rod (give spider the tape)
Red Rod (break rat-antnest with disk/tape)
Pink Rod (lizard man - yellow dock - need suit)
Brown Rod (give spacesuit for rod - to chief)
Violet Rod (move skeleton)
Silver Rod (look in ray gun barrel)
Clear Rod (look at laser with visor)
Green Rod ( turn on computer - need the square)
Blue Rod (disk under, basket on sphere, dial 14, teleport)
White Rod (lying loose in the drive bubble)
Entering Control Bubble (fire gun at bubble drive)
Proper Control Sequence (pink brown violet green blue)
Reachin Earth (proper orbit - "Galactic Overload")

That's it for this month. Don't forget, keep those hints and tips coming in. Address all correspondence to:-
"The Adventurer",
c/o The Secretary,
6 Arcot Close,
TARRO NSW
2322

# PRE-SCAN

BY

BRIAN RUTHERFORD

*What is meant by PRE-SCAN? we were asked by one of our members in last month's Newsletter. This month comes an answer from another of our members. Remember, if you have any problems, let the Editor know and he can 'arrange' for someone with the necessary knowledge to answer the question via the Newslettr.*

You know the time you have to wait after you have entered RUN and before the programme actually starts? Well, that is the time the computer takes to pre-scan the programme.

This pre-scan is when the computer establishes the necessary memory space for all the variables, arrays and data, plus sets up the look up tables for all the sub-programmes, both built in and user defined ones. The computer also makes sure that all the FOR NEXT loops match up, that is a NEXT for each FOR and a FOR for each NEXT & that the nesting of loops is correct.

In XB the programmer has the option of turning the pre-scan off with the command !@P- and to turn it on with the command !@P+. What use is this? you may ask. Well by setting up your programme in the right way and by judicious use of these commands the time taken to pre-scan the programme can be cut considerably.

As the main purpose of the pre-scan is to allocate memory for variables etc., the computer only needs to read the first reference to any of them. By following the guide on page seven of the addendum in the XB manual you can minimize the statements that need to be pre-scanned.

The rules being:

* Keep your first DATA statement in the pre-scan, for the pointer for the first READ to use.

* The first use of each variable and array, also include the OPTION BASE statement.

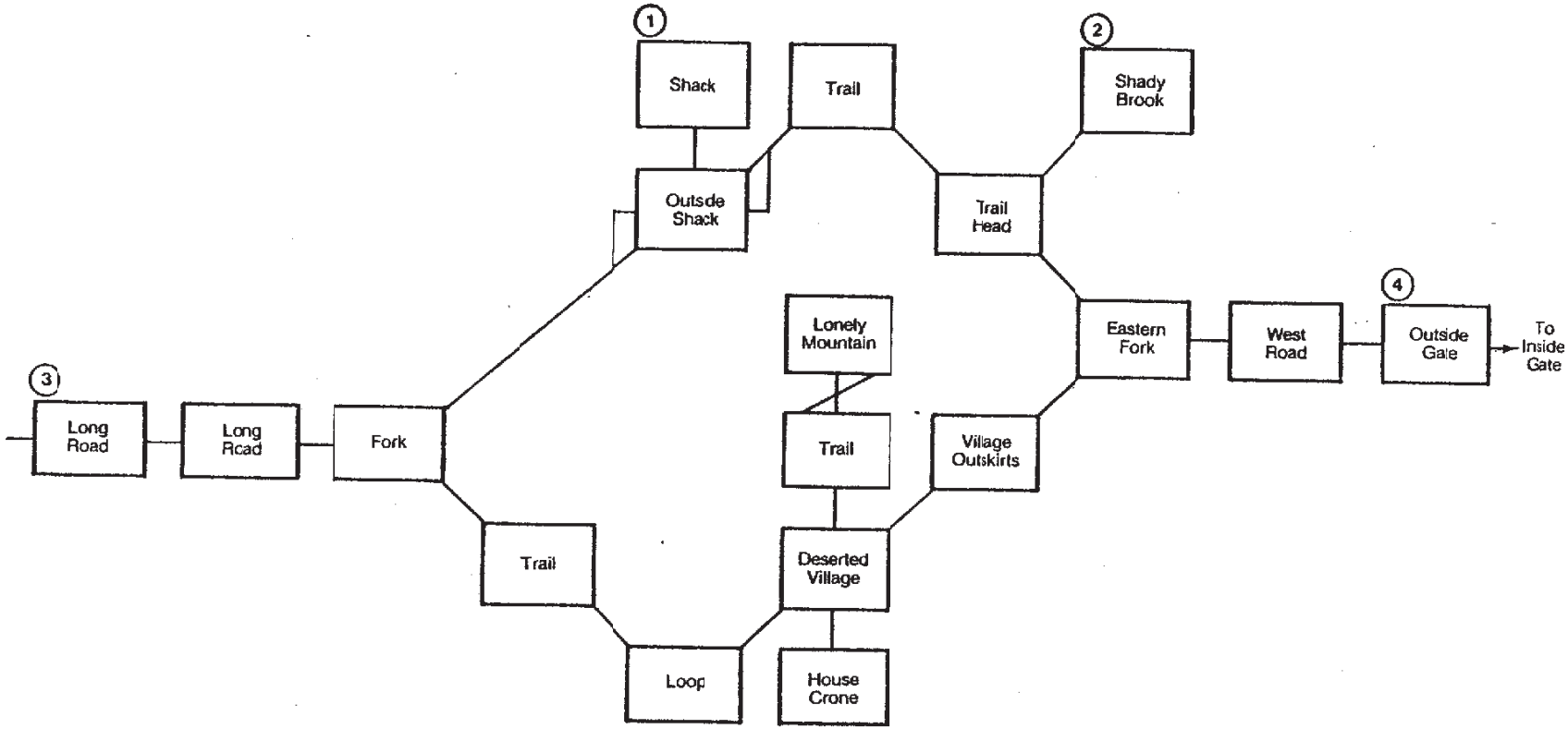
* Include the first reference to each CALL to any sub-programme.

* Include any DEF statements.

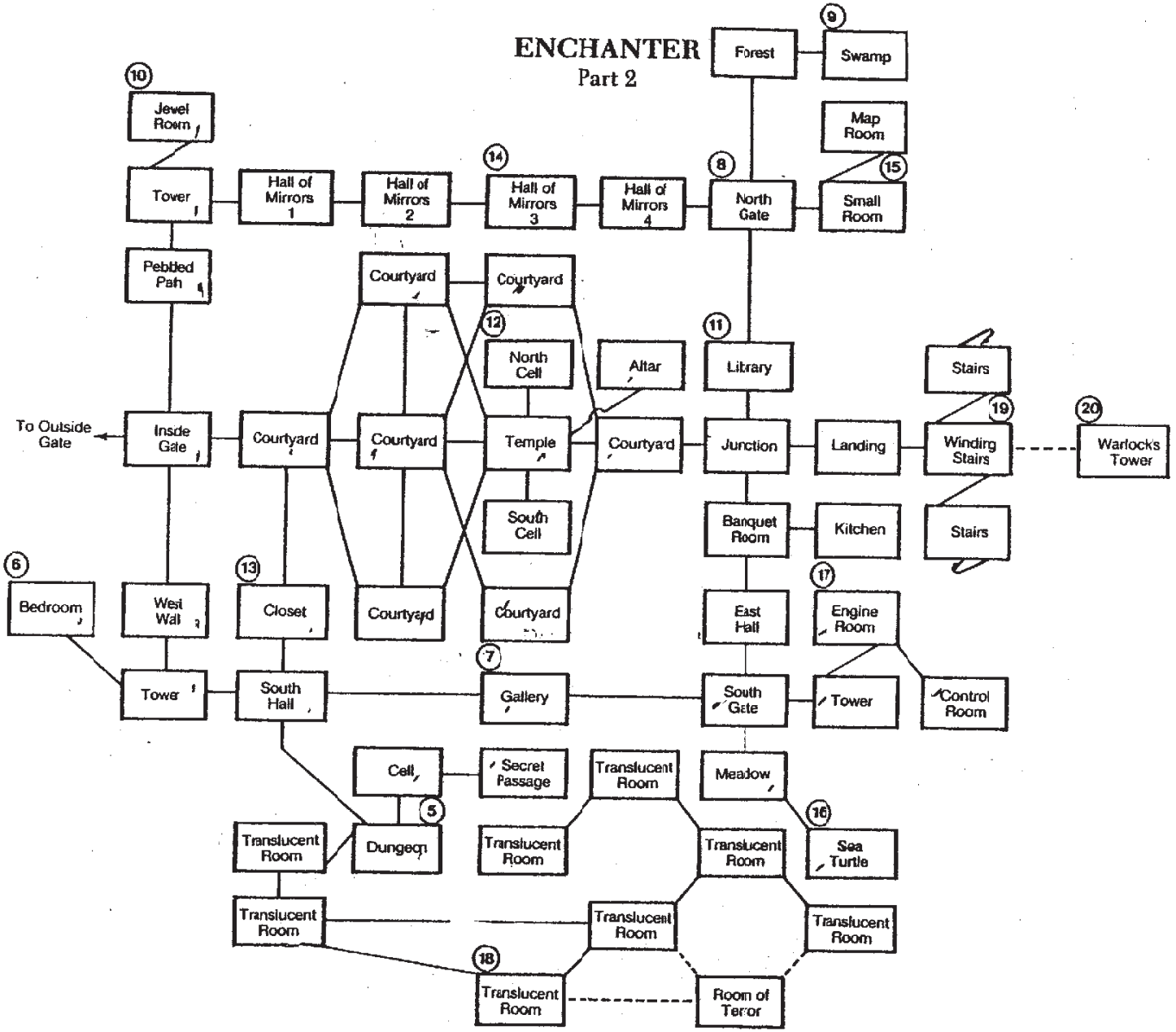
* Include all SUB and SUBEND statements of user defined sub-programmes so the computer knows where they start and end.

** Finally read pages 7-8-9-10 of the addendum to the XB manual.

# ENCHANTER
## Part 1



# ENCHANTER
## Part 2



# ASSEMBLY LANGUAGE

## FOR THE LAYMAN

WITH ALLAN WRIGHT, HV99ERS

Hello again, from what my wife calls the "tidy spot". I have often said to my wife that she is lucky that computing is my hobby. At least I'm usually about the house and when something needs to be done just a simple "HEY" can get me to do it. I think the reply was went like this; Ha, Ha, Ha!.

The good news is that the Federal Elections are out of the way which means a sharp decrease in the amount of double speak reported. Not that politicians have stopped, it just doesn't get the insane Media coverage after an Election. Even with that drawback a Free Democracy is far and away a better system than some of the Totalitarian Systems on offer around the world. Especially important and what we must protect is our freedom of speech and freedom to have opinions which we can express freely.

Last month I mentioned that I hoped to publish the source code of the exercise I had set the Assembly S.I.G. This was to be the code submitted by one of the Group members. Well! the last get together was held in the hub of the Hunter, Maitland at Pete Smith's house. Great night was had by all, 7:00pm to 1:00am debugging various versions of the exercise, some of which was NOT repeat NOT commented. The result is I have not got one to publish this month as my article.

DSR ERROR HANDLING.
During file access if the DSR detects an ERROR the second bit of the STATUS REGISTER is set. This allows us to use the code;

```
        BLWP  @DSRLNK
        DATA  8
        JEQ   ERR
```

to JUMP to an ERROR handling sub programme. Remember that to use JUMP the subprogramme entry point must be within >100 bytes of the current address. If it is outside this range then an alternative would be;

```
        BLWP  @DSRLNK
        DATA  8
        JNE   CONT
        BL    @ERR
        JMP   GETFIL
CONT    ---   ------
```

Whether this is good programming practise or not I will let others decided, I find this method of BRANCHING very useful indeed. The main attraction is that it allows the sub programme to be place anywhere, and it is LINKED for the return. A JUMP or BRANCH can take programme flow, for example back to the filename input routine.

WHEN IT GETS THERE.
The next thing to decide is what sort of ERROR handling you want to have in YOUR programme. My first thoughts on the matter were aimed at the User. With my self declared need to make programmes User friendly I considered that the User should be told as much as possible about the type of ERROR that has occurred. This in turn would allow the User to correct the situation with out much trauma, so long as he could read. So I started to write a super duper ERROR handling sub programme, after some considerable time I sat back and had a look at what I was doing and where I was headed.

The first thing which came to mind was the fact that with the programme I am currently writing I am getting a bit tight on for memory. The Error handler was starting to get fairly lengthy, especially taking into account the number of easy to understand TEXT messages I needed. Secondly I was spending quite some time programming but not on the actual programme.

What does the User need to know when an ERROR does occur. If he has just placed a disk into drive 2 and Typed in a filename to save some data to that disk. Should he need to be told that an ERROR has occurred in DISK DRIVE 2 and to have a look? Well I decided that to do this was going to be very expense memory wise and also take time which I could spend on the main body of the programme.

How does T.I. Writer report ERRORS? Surely I can use a similar reporting philosophy. In with the EDITOR and then force an ERROR. Ha! That will do me;

I/O ERROR CODE=Ø6

Out with the E/A manual and open to page 299 where the ERROR codes are listed. Alright showing that list in the programme instructions will be a lot less painful than coding them into the subprogramme I was writing.

In addition there is the fact that I have now restricted the size of the ERROR handler to a minimum length, saving a great swag of memory. Then on top of all this is the saving by decreasing the TEXT commitment from a prospective 6-7 messages down to 1 or 2.

AFTER THE EVENT.
The next consideration is what options should the User have available to retrieve the Error condition. Back to T.I. Writer Editor again, force the ERROR again. Press Enter and back to where I was prior to forcing the ERROR. At this point with T.I. Writer I can still escape back to the FUNNELWEB menu. That looked great to me.

I have allowed two options in my ERROR handler, the first is to RETRY. That takes programme flow back to the filename entry point with the cursor on a default filename ( the last one entered ). The second takes programme flow back to the MAIN menu.

ERROR REPORTING.
When DSR reports an ERROR by setting the EQUAL bit of the STATUS REGISTER the ERROR code is indicated in bits 1,2 and 3 of byte 1 of the PAB.

SOURCE CODE FOR ERROR SUBPROGRAMME.
**********

```
*THIS ROUTINE IS THE ERROR HANDLING FILE
ERR1   MOV  R11,@RETUR1      SAVE RET ADDR
       BL   @CLEAR1          CLEAR SCRN SUB PROG
       DATA 0,>2FF           AREA TO CLEAR
       LI   R0,PAB+1         ERROR CODE ADDR
       BLWP @VSBR            GET ERROR CODE
       SRL  R1,13            SHIFT TO LEAST SIGN BYTE
       CI   R1,5             IS IT EOF?
       JEQ  ERR3             IF SO GO TO MENU
       AI   R1,>30           ADD HEX 30
       LI   R0,>184          ADDRESS TO PRINT CODE NUMBER
       SWPB R1               PUT CODE INTO MOST SIGN BYTE
       BLWP @VSBW            WRITE CODE NUMBER TO SCRN
       LI   R0,>1A5          ADDRESS FOR MESSAGE
       LI   R1,ERMS6         ADDRESS OF TEXT
       LI   R2,>F            NO OF BYTES TO WRITE
       BLWP @VMBW            WRITE MESSAGE
       LI   R0,>2E1          ADDRESS OF SECOND MESSAGE
       LI   R1,MESS61        MESS FOR USER
       LI   R2,>1B           NUMBER OF BYTES
       BLWP @VMBW            WRITE MESSAGE
       CLR  @STATUS          CLEAR STATUS REG
ERR2   BLWP @KSCAN           GET KEY PRESS
       CB   @SPACE,@STATUS   ANY KEY PRESSED
       JNE  ERR2             NO THEN GO BACK
       CB   @KEYVAL,@ONE     IS ONE PRESSED
       JLT  ERR2             YES THEN BACK TO MENU
       CB   @KEYVAL,@ONE     CHECK FOR ONE
       JEQ  ERR3             GO BACK IF LESS THAN
       CB   @KEYVAL,@TWO     TWO PRESSED
       JGT  ERR2             IF GREATER THAN TWO GO BACK TO KSCAN
       MOV  @RETUR1,R11      RELOAD RETURN ADDRESS
       RT                    RETURN
ERR3   B    @MMENU2          BRANCH TO MAIN MENU
```

OPERATION DESCRIPTION.

```
ERR1   MOV   R11,@RETUR1
       BL    @CLEAR1
       DATA  0,>2FF
```

The LABEL ERR1 is the entry point for the sub programme. Since it was entered using BL the return address is placed in R11. This is MOV'ed into a word of memory set aside at LABEL RETUR1. The next two lines branch to my clear screen sub programme and the DATA specifies the area of screen to clear. In this case the whole screen.

```
       LI    RØ,PAB+1
       BLWP  @VSBR
```

The address in VDP RAM of the second byte on the PAB is placed in R0, the contents is returned in the most significant byte of R1.

```
        SRL   R1,13
        CI    R1,5
        JEQ   ERR3
```

The first three bits of R1 contain the ERROR code. The contents of R1 is moved to the right 13 places. As the move takes place vacated bits are filled with Ø's. Bits which move right in front of the data are lost forever into the bit bucket. If you want them you had better save them before SRL is used. The number remaining in R1 is now the error code. This is compared to 5. ERROR Code 5 is END OF FILE. When EOF is encountered the DSR closes the opened file for you. For example when reading a file into memory. If you want ALL the file loaded the programme can be allowed to continue accessing the DSR until EOF is reach. At this point an ERROR is detected. The subprogramme on seeing that the ERROR is EOF, Branches back to the MAIN menu

```
        AI    R1,>3Ø
        LI    RØ,>1B4
        BLWP  @VSBW
```

The contents of R1 now has >3Ø added to it to make it a displayable number. It is then written to screen position >1B4.

```
        LI    RØ,>1A5
        LI    R1,ERMSG
        LI    R2,>F
        BLWP  @VMBW
        LI    RØ,>2E1
        LI    R1,MESS61
        LI    R2,>18
        BLWP  @VMBW
```

Two messages are written to screen,

```
ERMSG  TEXT "I/O ERROR CODE="
MESS61 TEXT "ERROR 1-Menu 2-Try
again"
```

```
        CLR   @STATUS
ERR2    BLWP  @KSCAN
        CB    @SPACE,@STATUS
        JNE   ERR2
        CB    @KEYVAL,@ONE
        JLT   ERR2
        CB    @KEYVAL,@ONE
        JEQ   ERR3
        CB    @KEYVAL,@TWO
        JGT   ERR2
```

The STATUS REGISTER is cleared then the BRANCH to KSCAN is made, if a key is press the Third bit of STATUS is set, if not set the programme loops to ERR2. A BYTE directive is used for the LABEL SPACE BYTE >2Ø. The value of the key press is tested against ONE. If less than ONE the programme loops back to ERR2. If equal to ONE programme flow jumps to ERR3 and then back to the MAIN menu. If these conditions are not true the key press is tested against TWO. If greater than TWO programme flow returns to ERR2. The condition is now such that only a key press of the key TWO will allow the programme to RETURN to the calling programme.

```
        MOV   @RETUR1,R11
        RT
ERR3    B     @MMENU2
```

This last section of code wraps things up. The return address is MOV'ed back into R11 and then the BRANCH to that address is made. The last line BRANCHES back to the main menu after a key press of one is detected.

The only thing required is the place in your written instructions the ERROR Codes and their meanings.

OTHER ACCESSES.
All the discussions so far have dealt with the normal file handling features of the disc controller DSR. This is invoked by the well known;

```
        BLWP  @DSRLNK
        DATA  8
```

In addition to this feature the DSR ROM also has 7 built in sub programmes. The sub programmes in the DSR are accessed using the code;

```
        BLWP  @DSRLNK
        DATA  >A
```

If you read my article last month you will note that I said that to access these sub programmes DATA 1Ø should be used, that is not correct it is >A.

The sub programmes that are available are;

>1Ø SECTOR READ/WRITE
>11 DISC FORMATTING
>12 FILE WRITE PROTECTION

>13 FILE RENAME
>14 DIRECT FILE INPUT
>15 DIRECT FILE OUTPUT
>16 DISK BUFFER ALLOCATION

I am in the process of trying to learn the use of these sub programmes. If you are interested and would like to go further then I would recommend that you first read two very good articles on disc sector access by Wills McGovern in the JULY and SEPTEMBER 1986 H.V.NEWS. But be warned that some unexpected things can happen when working with these sub programmes unless you are extremely careful. In other words the usually qualification about my not being responsible for any damage do apply.

CLOSE.
One more interesting article which I would also recommend reading appeared in the APRIL/MAY issue of the MELBOURNE News Letter. Robin Bell has written and well documented the source code to load and run DM1000.

That's it from me, if the S.I.G. still have not come up with the goods next month then the article will have to be another surprise for you.

Until next month.
See Ya 'cross a keyboard.

Joe Wright.

# STOP PRESS

Many enquiries have been received during the past month regarding pricing and availability of the Mini Expansion System developed recently by TIsHUGs Peter Schubert and John Paine.

The following information on this was received by our Technical Co-ordinator Gary Jones from Peter Schubert on 6/8/87.

** The DISK CONTROLLER has double sided, double density (DSDD) capability and will control up to 4 floppy disk drives.
Cost of Disk Controller is $190.

** The MINI-PE MOTHERBOARD comes complete with EITHER, repeat EITHER ONE of the following :-
32K memory.
PIO (parallel printer port).
RS232 (serial port).
Cost of Motherboard with ONE of the above supplied is $85.00

** OPTIONS ON MOTHERBOARD.

| | | |
|---|---|---|
| RS232 | at | $50 |
| PIO | at | $50 |
| 32K | at | $50 |
| RS232/2 | at | $30 (2nd RS232 port) |

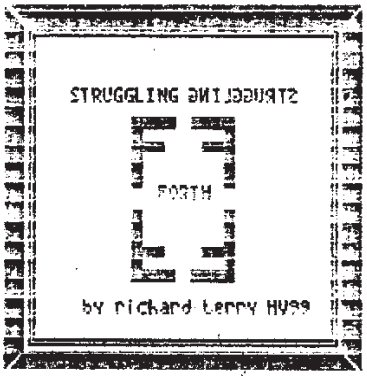
** These prices cover the completed circuit boards only and DOES NOT cover an enclosure to mount them in. A metal diecast box (painted and fitted) can be supplied for $35.00.

** AVAILABILITY.
The DISK CONTROLLER BOARD is NOW available through -
Peter Schubert (Phone~02-3585602)
PO.Box 28
KINGS CROSS, NSW.
2011
OR through the TIsHUG Shop -
Cyril Bohlsen on 02-6395947.

The MOTHERBOARD AND ITS OPTIONS have an expected delay of 3 to 4 weeks (from 6/8/87).

For further information on this contact Peter Schubert.

```
SCR #1
  0 ------------------------------------------------------------
  1 -                                                          -
  2 -  STRUGGLING FORTH                                        -
  3 -                                                          -
  4 -  AUGUST ARTICLE                                          -
  5 -                                                          -
  6 -                                                          -
  7 -                                                          -
  8 -  THE JOYS OF RAMMING FORTH                               -
  9 -  REQUEST FOR HELP WITH:                                  -
 10 -                : AUTOBOOT SCREENS                       -
 11 -                : I/O ERROR CHECKS                       -
 12 -                                                         -
 13 -                                                         -
 14 -                                                         -
 15 ------------------------------------------------------------
```




Well, its been some time since I put fingers to keyboard. As I predicted, I became so snowed under by winter work that my computing fell by the wayside for many months. Its only in the last 2 weeks I've resurrected the machine, and continued to Struggle Forth. I still intend writing the article I promised last time when I find the time. Apart from the fact I'd totally forgotten where I was program wise, reviewing where I left off reminded me of two frustrating problems with Forth I have yet to master - a problem with auto-booting, and I/O error checking. This article will not contain any program, but will describe the problems being experienced and ask for a help. I also thought I'd discuss the RAM-DISK and Forth, and a few odds and ends.

## THE JOYS OF RAMMING FORTH.

Many of you will already possess a RAM-DISK, either the HORIZON as in my case, or built from FISHUG boards. If you havn't, and you have a peripheral box, this is probably THE BEST peripheral card you can ever purchase for your computer. Installing this will allow you to prolong the useful life of your machine until some really good computers come on the market in a few years time.

The RAM-DISK is functionally identical to a floppy disk in a drive. You can initialise it, save any type of program or file to it, or back up from it. The difference is that because it is battery backed up RAM, its contents are not lost when you turn the machine off, so it loads things instantaneously when you power up. As there is no mechanical accessing (as in an ordinary floppy), and though a little slower its RAM can be thought of as an extension of the ordinary RAM in your computer. Hence a program like DM-1000 loads with barely a flicker on the screen. Like an ordinary floppy, it has its own disk index which is updated as you add or delete files, by the DOS. The disk size depends upon what you have, but is usually equivalent to a double sided single density of about 720 sectors.

You will remember how Forth doesn't use the normal system of files etc, and unless you write a basic header to it, a Forth disk isn't catalogable by the normal disk managers. Every four sectors of your RAM disk would correspond to 1K or 1 SCREEN on which to write your Forth code.

You can use the disk in several ways. You can keep it as a purely non Forth environment disk; soley a Forth work disk, not able to boot from, ie without the Forth core file perhaps configured as disk 3; or when configured as disk drive 1 and containing the core, the bootable Forth disk; or to house a combination of forth/basic/assembler programs.

For those of you owning a disk or contemplating it, you will have no difficulty setting it up for Forth alone, so lets just consider using

it as a mixed environment.

Set up the disk as say drive 3. Copy across whatever BASIC/ Assembler/files you want using DM-1000. Catalog the drive and note the last file on the list, or if you've got files from several disks, the last file you put on there. Next load DPATCH and using the 'search for a file' option find what the last sector used by the file is & write it down. All sectors above this will be unused.
Access the next sector, and write yourself a message here, such as "START OF FORTH TERRITORY", and re-write this using the FCTN 8.

Next load up Forth in the usual way. Calculate the screen this last sector will be on by dividing it by 4, and access this screen. Your message will be on this or the next screen. From this screen on you can use the RAM-DISK for either Forth source code or to keep compiled binary forth code to boot your Forth programs from.

A WORD OF CAUTION.
---

Three points to remember. Firstly, for the purposes of the DOS the RAM DISK is empty from the sector above the last file saved to the disk. If you are silly enough to save more files to the disk it will overwrite your Forth code, ie YOU may know you have Forth on the disk, but the system doesn't.

Secondly, as a corollary of this, if you are silly enough to CLEAR OR SCOPY OF SMOVE screens below the sector/screen you know to be your first permissable one for Forth, you will wipe out your non Forth files/programs.

Thirdly, be careful with the values on your boot screens of DISK-HI and DISK-SIZE. For instance though your drives may be DSDD for example, there may be times your are only using single sided/density disks. If you accidentally set these parameters wrong, you may inadvertently write over the lower part of your RAM-DISK, thinking you are about to access your second drive.

The message of all this is MAKE A BACKUP!!!!!!!! of your RAM-DISK.

Another comment. I have also inadvertently managed to write into the RAM-DISK operating system, is the ROS by being careless and poking things into the wrong spots of memory and had to completely re-load the ROS from the HORIZON system disk.

WAYS TO RAM FORTH.
---

Try CLEARing a group of screens on the ram-disk above your last sector, or MOVEing a group of screens. Fast isn't it. When your doing a lot of programming it makes life so much easier.

As a practical example I'll describe my system configuration. Drive 1,2 are DSDD drives with 360 screens. The RAM-DISK hence starts at screen 720. Accessing this will show you the start of the disk, with the diskname and first files. If you have only single sided drives, your values for DISK_SIZE AND DISK_HI will be different. To access your RAM disk on boot up you must include the correct values on your Boot screen - scrn 3. Examine screen 3 here as an example.

I have constructed a master menu which is binary saved and loads on boot-up, here pointed to on SCR# 852 and auto-started by RUN. Choosing from this then allows you to Binary load several programs which I use in the development of my programs, or the ordinary Forth environment. The quit option of each program again leads back to the master menu. The binary codes of all these, and the binary core of Forth itself ie the booting code, reside on the RAM-DISK itself. When you initially boot the system the only screen outside the RAM-DISK which need be accessed with a slow disk access time to an ordinary floppy is SCREEN 3. Once the initial boot takes place, all other master menu options load instantaneously, so many programs run as if merely sub-options of one whole program.

TI-FORTH BOOT DRIVE>2,3
---
It is possible to boot ordinary TI-FORTH from drives other than No.1 Yesterday I quickly experimented with this. Using DM-1000 copy across FORTH and FORTHSAVE to your RAM-DISK. Using DPatch find the

start sector of FORTH (DV80FILE) and at Byte no. 22 change the 1 to whichever drive you want to call it from, and then using Function 8, rewrite the sector to disk. When booting Forth this way you must still have a disk in drive 1, but it need only have an operational screen 3, as this is accessed once the core is loaded. I had no trouble in booting a program written in TI-Forth this way, but let me know if any of you experience difficulty.

BINARY LOADING.
---

As an observation of doing BLOADS, I was always under the impression one had to FORGET an application before bringing in another "on top" of the old one. After some experimentation and reading how BSAVE saves things, it seems to me that when changing applications one can just dump the new overlay completely over the top of the old, as it brings its own parameters with it.

NOW THE PROBLEMS.
---

When I ceased actively programming several months ago, I was being frustrated in two main areas. Again I make a plea for help from anyone out there with a deeper understanding of Forth than I, to make the effort to write and help us over here.

PROBLEM 1:AUTOBOOTING.
---

When autobooting a program, ie making the final definition of the application the last word on the screen being booted (see scrn#3), I have had a lot of problems with the program locking up at any point within it where data in the form of strings is being accepted. Eg if I accepted a key inputted for a number using our GET$ routine, as detailed in previous magazine articles, the program locks into an interminable cycle at this point. If I don't make the program an auto-boot one, ie binary load the code, then in the immediate mode, not on a screen type the key word to run the program, this type of data accepting works with no problems. When data is accepted using a straight KEY routine for example getting a number between 1-9 as : def KEY DUP EMIT 48 - ; with no validation, or validation in a BEGIN...UNTIL OR MYSELF loop, it works fine, and autoboots. As soon as I put error checking and use either MYSELF or BEGIN... UNTIL with string accepts, to continue the process until the correct flag has been set, it locks up on auto-boot but runs fine in the immediate mode.

If anyone out there is prepared to help I'll post them a disk with the listing and program to peruse.

PROBLEM 2:I/O ERROR CHECKING.
---

This is not so much of a problem as we've nearly sussed it out, and Joe assures me he's au fait with it in assembler. However, if anyone out there including

*** KEITH BRUCE ***

feels like writing a clear comprehensive article on it .....? Well, you know where to post it.

ADDRESS FOR CORRESPONDENCE.
---

RICHARD TERRY
141 DUDLEY RD
WHITEBRIDGE 2290
NEWCASTLE
AUSTRALIA
(049) 436861/22450

```
SCR #3
  0 ( CONFIGURATION & STARTUP SCREEN )
  1 360 DISK_SIZE ! 900 DISK_HI !     ( CONFIGURE DISKS              )
  2    822 LOAD                       ( DESCENDER CHARACTER SET      )
  3 16 SYSTEM  823 BLOCK 0 227  2 SYSTEM
  4 HEX 10 83C2 C! DEC                ( QUIT OFF                     )
  5
  6    800 BLOAD HERE FENCE           ( LOAD KERNEL EXTENSION        )
  7    910 BLOAD                      ( LOAD TASK 1 WORDS            )
  8    815 BLOAD ( DEV40)             ( LOAD TASK 2 EDITOR & DEV     )
  9    821 BLOAD                      ( LOAD STRING WORDS ETC        )
 10
 11 1 VDPMDE !                        ( SET VDP MODE )
 12 CHAR$
 13 : \   IN @ 64 / 1+ 64 * IN ! ; IMMEDIATE  ( ignore rest line  )
 14 SET-PTR PIO
 15 852 BLOAD  RUN
```

# SAVING DISK SPACE

## by

## gary jones

When designing a banner printing programme recently there was a need to store, on disk, 3 TRIPLE DIGIT values for each input and possibly upto 160 inputs needed to construct each character.With 41 characters in 2 different FONTS to be designed, this was going to consume a large amount of disk space.

To calculate the number of disk sectors per character a small programme was devised. The programme opens a file named "TESTFILE" and loads, onto disk, 160 inputs which are 3 sets of randomly generated triple digit values(100-199).

```
10 OPEN #1:"DSK1.TESTFILE",
OUTPUT,INTERNAL
20 FOR I=1 TO 160
30 A,B,C=INT(100*RND)+100
40 PRINT #1:A,B,C
50 NEXT I::CLOSE #1
```

After running the programme and reading the number of sectors used on the disk, it was my conclusion that 19 sectors was by far too many per character to allow all the characters and printing programmes to be stored on a SSSD disk.

After referring to the URG II-132, a string variable will use less bytes than a numeric variable, so I then changed the above programme to the following!-

```
10 OPEN #1:"DSK1.TESTFILE",
OUTPUT,INTERNAL
20 FOR I=1 TO 160
30 A=INT(100*RND)+100
40 A$,B$,C$=STR$(A)
50 PRINT #1:A$,B$,C$
60 NEXT I::CLOSE #1
```

Even with this small change to the programme, the number of sectors was reduced by half, to 9. I still considered this to be higher than the allowable number of sectors per character.

The next step was to come up with some way of writing to disk a single character string variable which replaces the numeric value. The result is the following:-

```
10 OPEN #1:"DSK1.TESTFILE",
OUTPUT,INTERNAL
20 FOR I=1 TO 160
30 A=INT(100*RND)+100
40 A$,B$,C$=CHR$(A)
50 PRINT #1:A$,B$,C$
60 NEXT A::CLOSE #1
```

When the number of sectors where checked, the number of 6 was well within the limit I had proposed.

The next question to be answered was, how is a single character string variable converted back into a numeric value ? Note the following programme:-

```
10 OPEN #1:"DSK1.TESTFILE",
INPUT,INTERNAL
20 FOR I=1 TO 160
30 INPUT #1:A$,B$,C$
40 PRINT ASC(A$)
50 NEXT I::CLOSE #1
```

The only restriction is, the numeric value stored on a file can be no larger then "255", as data corruption will occur in the string variables being filed over this number. Therefore the range of values capable of being stored is 0-255.

I hope the above programmes will shed a little light on how file-handling works and also will allow you to sector crunch large numeric files.

C#97 Current

Mr.R. CARMANY
1504 Larson St.
GREENSBORO NC.27407
U.S.A.